

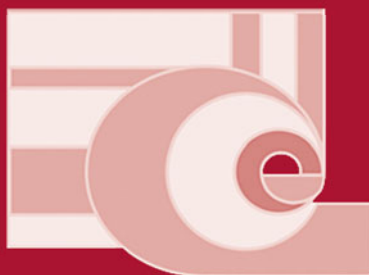
Alexander Gelbukh (Ed.)

LNCS 13397

Computational Linguistics and Intelligent Text Processing

19th International Conference, CILing 2018
Hanoi, Vietnam, March 18–24, 2018
Revised Selected Papers, Part II

2
Part II



 Springer

Lecture Notes in Computer Science

13397

Founding Editors

Gerhard Goos
Juris Hartmanis

Editorial Board Members

Elisa Bertino, *Purdue University, West Lafayette, IN, USA*

Wen Gao, *Peking University, Beijing, China*

Bernhard Steffen , *TU Dortmund University, Dortmund, Germany*

Moti Yung , *Columbia University, New York, NY, USA*

The series Lecture Notes in Computer Science (LNCS), including its subseries Lecture Notes in Artificial Intelligence (LNAI) and Lecture Notes in Bioinformatics (LNBI), has established itself as a medium for the publication of new developments in computer science and information technology research, teaching, and education.


LNCS enjoys close cooperation with the computer science R & D community, the series counts many renowned academics among its volume editors and paper authors, and collaborates with prestigious societies. Its mission is to serve this international community by providing an invaluable service, mainly focused on the publication of conference and workshop proceedings and postproceedings. LNCS commenced publication in 1973.

Alexander Gelbukh
Editor

Computational Linguistics and Intelligent Text Processing

19th International Conference, CICLing 2018
Hanoi, Vietnam, March 18–24, 2018
Revised Selected Papers, Part II

Editor

Alexander Gelbukh 
Instituto Politécnico Nacional
Mexico City, Mexico

ISSN 0302-9743

ISSN 1611-3349 (electronic)

Lecture Notes in Computer Science

ISBN 978-3-031-23803-1

ISBN 978-3-031-23804-8 (eBook)

<https://doi.org/10.1007/978-3-031-23804-8>

© Springer Nature Switzerland AG 2023

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors, and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

CICLing 2019 was the 20th International Conference on Computational Linguistics and Intelligent Text Processing. The CICLing conferences provide a wide-scope forum for discussion of the art and craft of natural language processing research, as well as the best practices in its applications.

This set of two books contains three invited papers and a selection of regular papers accepted for presentation at the conference. Since 2001, the proceedings of the CICLing conferences have been published in Springer’s Lecture Notes in Computer Science series as volumes 2004, 2276, 2588, 2945, 3406, 3878, 4394, 4919, 5449, 6008, 6608, 6609, 7181, 7182, 7816, 7817, 8403, 8404, 9041, 9042, 9623, 9624, 10761, 10762, 13396, and 13397.

The set has been structured into 14 sections representative of the current trends in research and applications of natural language processing: General; Information Extraction; Information Retrieval; Language Modeling; Lexical Resources; Machine Translation; Morphology, Syntax, Parsing; Name Entity Recognition; Semantics and Text Similarity; Sentiment Analysis; Speech Processing; Text Categorization; Text Generation; and Text Mining.

In 2019 our invited speakers were Preslav Nakov (Qatar Computing Research Institute, Qatar), Paolo Rosso (Universidad Politécnic de Valencia, Spain), Lucia Specia (University of Sheffield, UK), and Carlo Strapparava (Fondazione Bruno Kessler, Italy). They delivered excellent extended lectures and organized lively discussions. Full contributions of these invited talks are included in this book set.

After a double-blind peer review process, the Program Committee selected 95 papers for presentation, out of 335 submissions from 60 countries.

To encourage authors to provide algorithms and data along with the published papers, we selected three winners of our Verifiability, Reproducibility, and Working Description Award. The main factors in choosing the awarded submission were technical correctness and completeness, readability of the code and documentation, simplicity of installation and use, and exact correspondence to the claims of the paper. Unnecessary sophistication of the user interface was discouraged; novelty and usefulness of the results were not evaluated, instead they were evaluated for the paper itself and not for the data.

The following papers received the Best Paper Awards, the Best Student Paper Award, as well as the Verifiability, Reproducibility, and Working Description Awards, respectively:

Best Verifiability, Reproducibility, and Working Description Award: “Text Analysis of Resumes and Lexical Choice as an Indicator of Creativity”, Alexander Rybalov.

Best Student Paper Award: “Look Who’s Talking: Inferring Speaker Attributes from Personal Longitudinal Dialog”, Charles Welch, Veronica Perez-Rosas, Jonathan Kummerfeld, Rada Mihalcea.

Best Presentation Award: “A Framework to Build Quality into Non-expert Translations”, Christopher G. Harris.

Best Poster Award, Winner (Shared): “Sentiment Analysis Through Finite State Automata”, Serena Pelosi, Alessandro Maisto, Lorenza Melillo, and Annibale Elia. And “Toponym Identification in Epidemiology Articles: A Deep Learning Approach”, Mohammad Reza Davari, Leila Kosseim, Tien D. Bui.

Best Inquisitive Mind Award: Given to the attendee who asked the most (good) questions to the presenters during the conference, Natwar Modani.

Best Paper Award, First Place: “Contrastive Reasons Detection and Clustering from Online Polarized Debates”, Amine Trabelsi, Osmar Zaiane.

Best Paper Award, Second Place: “Adversarial Training based Cross-lingual Emotion Cause Extraction”, Hongyu Yan, Qinghong Gao, Jiachen Du, Binyang Li, Ruifeng Xu.

Best Paper Award, Third Place (Shared): “EAGLE: An Enhanced Attention-Based Strategy by Generating Answers from Learning Questions to a Remote Sensing Image”, Yeyang Zhou, Yixin Chen, Yimin Chen, Shunlong Ye, Mingxin Guo, Ziqi Sha, Heyu Wei, Yanhui Gu, Junsheng Zhou, Weiguang Qu.

Best Paper Award, Third Place (Shared): “dpUGC: Learn Differentially Private Representation for User Generated Contents”, Xuan-Son Vu, Son Tran, Lili Jiang.

A conference is the result of the work of many people. First of all, I would like to thank the members of the Program Committee for the time and effort they devoted to the reviewing of the submitted articles and to the selection process. Obviously, I thank the authors for their patience in the preparation of the papers, not to mention the development of the scientific results that form this book. I also express my most cordial thanks to the members of the local Organizing Committee for their considerable contribution to making this conference become a reality.

November 2022

Alexander Gelbukh

Organization

CICLing 2019 (20th International Conference on Computational Linguistics and Intelligent Text Processing) was hosted by the University of La Rochelle (ULR), France, and organized by the L3i laboratory of the University of La Rochelle (ULR), France, in collaboration with the Natural Language and Text Processing Laboratory of the CIC, IPN, the Mexican Society of Artificial Intelligence (SMIA), and the NewsEye project. The NewsEye project received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 770299.

The conference aims to encourage the exchange of opinions between the scientists working in different areas of the growing field of computational linguistics and intelligent text and speech processing.

Program Chair

Alexander Gelbukh

Instituto Politécnico Nacional, Mexico

Organizing Committee

Antoine Doucet (Chair)

University of La Rochelle, France

Nicolas Sidère (Co-chair)

University of La Rochelle, France

Cyrille Suire (Co-chair)

University of La Rochelle, France

Members

Karell Bertet

L3i Laboratory, University of La Rochelle, France

Mickaël Coustaty

L3i Laboratory, University of La Rochelle, France

Salah Eddine

L3i Laboratory, University of La Rochelle, France

Christophe Rigaud

L3i Laboratory, University of La Rochelle, France

Additional Support

Viviana Beltran

L3i Laboratory, University of La Rochelle, France

Jean-Loup Guillaume

L3i Laboratory, University of La Rochelle, France

Marwa Hamdi

L3i Laboratory, University of La Rochelle, France

Ahmed Hamdi

L3i Laboratory, University of La Rochelle, France

Nam Le

L3i Laboratory, University of La Rochelle, France

Elvys Linhares Pontes

L3i Laboratory, University of La Rochelle, France

Muzzamil Luqman	L3i Laboratory, University of La Rochelle, France
Zuheng Ming	L3i Laboratory, University of La Rochelle, France
Hai Nguyen	L3i Laboratory, University of La Rochelle, France
Armelle Prigent	L3i Laboratory, University of La Rochelle, France
Mourad Rabah	L3i Laboratory, University of La Rochelle, France

Program Committee

Alexander Gelbukh	Instituto Politécnico Nacional, Mexico
Leslie Barrett	Bloomberg, USA
Leila Kosseim	Concordia University, Canada
Aladdin Ayesh	De Montfort University, UK
Srinivas Bangalore	Interactions, USA
Ivandre Paraboni	University of São Paulo, Brazil
Hermann Moisl	Newcastle University, UK
Kais Haddar	MIRACL Laboratory, Faculté des Sciences de Sfax, Tunisia
Cerstin Mahlow	ZHAW Zurich University of Applied Sciences, Switzerland
Alma Kharrat	Microsoft, USA
Dafydd Gibbon	Bielefeld University, Germany
Evangelos Milios	Dalhousie University, Canada
Kjetil Nørvåg	Norwegian University of Science and Technology, Norway
Grigori Sidorov	CIC-IPN, Mexico
Hiram Calvo	Nara Institute of Science and Technology, Japan
Piotr W. Fuglewicz	TiP, Poland
Aminul Islam	University of Louisiana at Lafayette, USA
Michael Carl	Kent State University, USA
Guillaume Jacquet	Joint Research Centre, EU
Suresh Manandhar	University of York, UK
Bente Maegaard	University of Copenhagen, Denmark
Tarik Kişla	Ege University, Turkey
Nick Campbell	Trinity College Dublin, Ireland
Yasunari Harada	Waseda University, Japan
Samhaa El-Beltagy	Newgiza University, Egypt
Anselmo Peñas	NLP & IR Group, UNED, Spain
Paolo Rosso	Universitat Politècnica de València, Spain
Horacio Rodriguez	Universitat Politècnica de Catalunya, Spain
Yannis Haralambous	IMT Atlantique & UMR CNRS 6285 Lab-STICC, France
Niladri Chatterjee	IIT Delhi, India

Manuel Vilares Ferro	University of Vigo, Spain
Eva Hajicova	Charles University, Prague, Czech Republic
Preslav Nakov	Qatar Computing Research Institute, HBKU, Qatar
Bayan Abushawar	Arab Open University, Jordan
Kemal Oflazer	Carnegie Mellon University in Qatar, Qatar
Hatem Haddad	iCompass, Tunisia
Constantin Orasan	University of Wolverhampton, UK
Masaki Murata	Tottori University, Japan
Efstathios Stamatatos	University of the Aegean, Greece
Mike Thelwall	University of Wolverhampton, UK
Stan Szpakowicz	University of Ottawa, Canada
Tunga Gungor	Bogazici University, Turkey
Dunja Mladenic	Jozef Stefan Institute, Slovenia
German Rigau	IXA Group, UPV/EHU, Spain
Roberto Basili	University of Roma Tor Vergata, Italy
Karin Harbusch	University Koblenz-Landau, Germany
Elena Lloret	University of Alicante, Spain
Ruslan Mitkov	University of Wolverhampton, UK
Viktor Pekar	University of Birmingham, UK
Attila Novák	Pázmány Péter Catholic University, Hungary
Horacio Saggion	Universitat Pompeu Fabra, Spain
Soujanya Poria	Nanyang Technological University, Singapore
Rada Mihalcea	University of North Texas, USA
Partha Pakray	National Institute of Technology Silchar, India
Alexander Mehler	Goethe-University Frankfurt am Main, Germany
Octavian Popescu	IBM, USA
Hitoshi Isahara	Toyohashi University of Technology, Japan
Galia Angelova	Institute for Parallel Processing, Bulgarian Academy of Sciences, Bulgaria
Pushpak Bhattacharyya	IIT Bombay, India
Farid Meziane	University of Derby, UK
Ales Horak	Masaryk University, Czech Republic
Nicoletta Calzolari	Istituto di Linguistica Computazionale – CNR, Italy
Milos Jakubicek	Lexical Computing, UK
Ron Kaplan	Nuance Communications, USA
Hassan Sawaf	Amazon, USA
Marta R. Costa-Jussà	Institute for Infocomm Research, Singapore
Sivaji Bandyopadhyay	Jadavpur University, India
Yorrick Wilks	University of Sheffield, UK
Vasile Rus	University of Memphis, USA

Christian Boitet
Khaled Shaalan
Philipp Koehn

Université Grenoble Alpes, France
The British University in Dubai, UAE
Johns Hopkins University, USA

Software Reviewing Committee

Ted Pedersen
Florian Holz
Miloš Jakubíček
Sergio Jiménez Vargas
Miikka Silfverberg
Ronald Winnemöller

Best Paper Award Selection Committee

Alexander Gelbukh
Eduard Hovy
Rada Mihalcea
Ted Pedersen
Yorick Wilks

Contents – Part II

Semantics and Text Similarity

Finding Word Sense Embeddings of Known Meaning	3
<i>Lyndon White, Roberto Togneri, Wei Liu, and Mohammed Bennamoun</i>	
Using Shallow Semantic Analysis to Implement Automated Quality Assessment of Web Health Care Information	17
<i>Yanjun Zhang, Robert E. Mercer, Jacquelyn Burkell, and Hong Cui</i>	
OntoSenseNet: A Verb-Centric Ontological Resource for Indian Languages	32
<i>Jyoti Jha, Sreekavitha Parupalli, and Navjyoti Singh</i>	
Detection of Change in the Senses of AI in Popular Discourse	46
<i>Ahmet Suerdem, Tugba Dalyan, and Savaş Yıldırım</i>	
Sparse Word Representation for RNN Language Models on Cellphones	59
<i>Chong Ruan and Yanshuang Liu</i>	

Sentiment Analysis

Predicting Email Opens with Domain-Sensitive Affect Detection	71
<i>Niyati Chhaya, Kokil Jaidka, and Rahul Wadbude</i>	
Detection of Suicidal Intentions of Tunisians via Facebook	80
<i>Jihen Nasri and Salma Jamoussi</i>	
Relationships and Sentiment Analysis of Fictional or Real Characters	92
<i>Paul Diac, Cătălina Mărănduc, and Mihaela Colhon</i>	
Sentiment Analysis of Code-Mixed Languages Leveraging Resource Rich Languages	104
<i>Narendra Choudhary, Rajat Singh, Ishita Bindlish, and Manish Shrivastava</i>	
<i>Emotions Are Universal: Learning Sentiment Based Representations of Resource-Poor Languages Using Siamese Networks</i>	115
<i>Narendra Choudhary, Rajat Singh, Ishita Bindlish, and Manish Shrivastava</i>	

Contrastive Learning of Emoji-Based Representations for Resource-Poor Languages	129
<i>Narendra Choudhary, Rajat Singh, Ishita Bindlish, and Manish Shrivastava</i>	
Aspect-Sentiment Embeddings for Company Profiling and Employee Opinion Mining	142
<i>Rajiv Bajpai, Devamanyu Hazarika, Kunal Singh, Sruthi Gorantla, Erik Cambria, and Roger Zimmermann</i>	
A Decision-Level Approach to Multimodal Sentiment Analysis	161
<i>Haithem Afli, Jason Burns, and Andy Way</i>	
Text-Image Sentiment Analysis	169
<i>Qian Chen, Edoardo Ragusa, Iti Chaturvedi, Erik Cambria, and Rodolfo Zunino</i>	
A Study on Far-Field Emotion Recognition Based on Deep Convolutional Neural Networks	181
<i>Panikos Heracleous, Yasser Mohammad, Koichi Takai, Keiji Yasuda, Akio Yoneyama, and Fumiaki Sugaya</i>	
Syntax and Parsing	
Cross-Framework Evaluation for Portuguese POS Taggers and Parsers	197
<i>Sandra Collovini, Henrique D. P. Santos, Thiago Lima, Evandro Fonseca, Bolivar Pereira, Marlo Souza, Sílvia Moraes, and Renata Vieira</i>	
Parsing Arabic with a Semi-automatically Generated TAG: Dealing with Linguistic Phenomena	209
<i>Cherifa Ben Khelil, Chiraz Ben Othmane Zribi, Denys Duchier, and Yannick Parmentier</i>	
Recognizing Textual Entailment Using Weighted Dependency Relations	221
<i>Tanik Saikh, Sudip Kumar Naskar, and Asif Ekbal</i>	
Hypernymy Detection for Vietnamese Using Dynamic Weighting Neural Network	234
<i>Van-Tan Bui, Phuong-Thai Nguyen, and Van-Lam Pham</i>	
Arbobanko - A Treebank for Esperanto	248
<i>Eckhard Bick</i>	

Sentence Compression as a Supervised Learning with a Rich Feature Space 261
Elena Churkin, Mark Last, Marina Litvak, and Natalia Vanetik

Text Categorization and Clustering

Multilabel Text Classification of Unbalanced Datasets: Two-Pass NNMF 275
Gabriella Skitalinskaya and John Cardiff

Algorithmic Segmentation of Job Ads Using Textual Analysis 287
*Benjamin Murauer, Michael Tschuggnall, Günther Specht,
 and Julia Brandl*

Neural Network Architecture for Credibility Assessment of Textual
 Claims (Best Paper Award, First Place) 301
*Narendra Choudhary, Rajat Singh, Ishita Bindlish,
 and Manish Shrivastava*

SMGKM: An Efficient Incremental Algorithm for Clustering Document
 Collections 314
*Adil Bagirov, Sattar Seifollahi, Massimo Piccardi,
 Ehsan Zare Borzeshi, and Bernie Kruger*

From Emoji Usage to Categorical Emoji Prediction 329
Gaël Guibon, Magalie Ochs, and Patrice Bellot

Text Generation

Towards Social Context Summarization with Convolutional Neural
 Networks 341
Minh-Tien Nguyen, Duc-Vu Tran, Viet-Anh Phan, and Le-Minh Nguyen

Towards Event Timeline Generation from Vietnamese News 354
Van-Chung Vu, Thi-Thanh Ha, and Kiem-Hieu Nguyen

An Abstractive Text Summarization Using Recurrent Neural Network 364
Dipanwita Debnath, Partha Pakray, Ranjita Das, and Alexander Gelbukh

Text Mining

Using Reviewer Information to Improve Performance of Low-Quality
 Review Detection 381
Qingliang Miao, Changjian Hu, and Feiyu Xu

Advanced Text Mining Methods for Bilingual Lexicon Extraction from Specilized Comparable Corpora	400
<i>Sourour Belhaj Rhouma, Chiraz Latiri, and Catherine Berrut</i>	
Classifiers for Yelp-Reviews Based on GMDH-Algorithms	412
<i>Mikhail Alexandrov, Gabriella Skitalinskaya, John Cardiff, Olexiy Koshulko, and Elena Shushkevich</i>	
Research Collaboration Analysis Using Text and Graph Features	431
<i>Drahomira Herrmannova, Petr Knoth, Christopher Stahl, Robert Patton, and Jack Wells</i>	
Aspect Specific Opinion Expression Extraction Using Attention Based LSTM-CRF Network	442
<i>Abhishek Laddha and Arjun Mukherjee</i>	
Author Index	455

Contents – Part I

General

Combining Graph-Based Dependency Features with Convolutional Neural Network for Answer Triggering	3
<i>Deepak Gupta, Sarah Kohail, and Pushpak Bhattacharyya</i>	
Prediction of Cryptocurrency Market	17
<i>Rareş Chelmuş, Daniela Gîfu, and Adrian Iftene</i>	
I-vectors and Deep Convolutional Neural Networks for Language Identification in Clean and Reverberant Environments	30
<i>Panikos Heracleous, Yasser Mohammad, Kohichi Takai, Keiji Yasuda, and Akio Yoneyama</i>	
Arabic Named Entity Recognition Using Clustered Word Embedding	41
<i>Caroline Sabty, Mohamed Elmahdy, and Slim Abdennadher</i>	
Connecting the Content of Books to the Web and the Real World	50
<i>Dan Cristea, Ionuț Pistol, and Daniela Gîfu</i>	
Finding Questions in Medical Forum Posts Using Sequence Labeling Approach	62
<i>Adrianus Saga Ekakristi, Rahmad Mahendra, and Mirna Adriani</i>	
Argumentation in Text: Discourse Structure Matters	74
<i>Boris Galitsky, Dmitry Ilvovsky, and Dina Pisarevskaya</i>	
A Deep Learning Approach for Multimodal Deception Detection	87
<i>Gangeshwar Krishnamurthy, Navonil Majumder, Soujanya Poria, and Erik Cambria</i>	
Author Profiling and Authorship Attribution, Social Network Analysis	
Hamtajoo: A Persian Plagiarism Checker for Academic Manuscripts	99
<i>Vahid Zarrabi, Salar Mohtaj, and Habibollah Asghari</i>	
Analyzing Stylistic Variation Across Different Political Regimes	110
<i>Liviu P. Dinu and Ana Sabina Uban</i>	

Towards Automated *Fiqh* School Authorship Attribution 124
Maha Al-Yahya

Stance and Gender Detection in Spanish Tweets 131
José-Ángel González, Lluís-F. Hurtado, and Ferran Pla

Information Retrieval, Information Extraction

Natural-Annotation-Based Malay Multiword Expressions Extraction
and Clustering 143
Wuying Liu and Lin Wang

Self-adaptive Privacy Concern Detection for User-Generated Content 153
Xuan-Son Vu and Lili Jiang

Complexity of Russian Academic Texts as the Function of Syntactic
Parameters 168
*Valery Solovyev, Marina Solnyshkina, Vladimir Ivanov,
and Svetlana Timoshenko*

An Experimental Approach for Information Extraction in Multi-party
Dialogue Discourse 180
*Pegah Alizadeh, Peggy Cellier, Thierry Charnois, Bruno Crémilleux,
and Albrecht Zimmermann*

Automatic Matching and Expansion of Abbreviated Phrases Without
Context 193
*Chloé Artaud, Antoine Doucet, Vincent Poulain D’Andecy,
and Jean-Marc Ogier*

Lexical Resources

Relation Extraction Datasets in the Digital Humanities Domain and Their
Evaluation with Word Embeddings 207
*Gerhard Wohlgenannt, Ekaterina Chernyak, Dmitry Ilvovsky,
Ariadna Barinova, and Dmitry Mouromtsev*

One World - Seven Thousand Languages (Best Paper Award, Third Place) 220
Fausto Giunchiglia, Khuyagbaatar Batsuren, and Abed Alhakim Freihat

Analysis of Speeches in Indian Parliamentary Debates 236
Sakala Venkata Krishna Rohit and Navjyoti Singh

Enrichment of OntoSenseNet: Adding a Sense-annotated Telugu Lexicon	250
<i>Sreekavitha Parupalli and Navjyoti Singh</i>	

Machine Translation

A Neural Network Classifier Based on Dependency Tree for English-Vietnamese Statistical Machine Translation	265
<i>Viet Hong Tran, Quan Hoang Nguyen, and Vinh Van Nguyen</i>	

The Utility of Hierarchical Phrase-Based Model Machine Translation for Low Resource Languages	279
<i>S. Yashothara and R. T. Uthayasanker</i>	

Automatic Method to Build a Dictionary for Class-Based Translation Systems	289
<i>Kohichi Takai, Gen Hattori, Keiji Yasuda, Panikos Heracleous, Akio Ishikawa, Kazunori Matsumoto, and Fumiaki Sugaya</i>	

Addressing the Issue of Unavailability of Parallel Corpus Incorporating Monolingual Corpus on PBSMT System for English-Manipuri Translation	299
<i>Amika Achom, Partha Pakray, and Alexander Gelbukh</i>	

On the Influence of Machine Translation on Language Origin Obfuscation	320
<i>Benjamin Murauer, Michael Tschuggnall, and Günther Specht</i>	

Error Classification and Evaluation of Machine Translation Evaluation Metrics for Hindi as a Target Language	331
<i>Samiksha Tripathi and Vineet Kansal</i>	

Morphology, Syntax

Unsupervised Learning of Word Segmentation: Does Tone Matter?	349
<i>Pierre Godard, Kevin Löser, Alexandre Allauzen, Laurent Besacier, and François Yvon</i>	

POS, ANA and LEM: Word Embeddings Built from Annotated Corpora Perform Better (Best Paper Award, Second Place)	360
<i>Attila Novák and Borbála Novák</i>	

Automatic Normalization of Word Variations in Code-Mixed Social Media Text	371
<i>Rajat Singh, Nurendra Choudhary, and Manish Shrivastava</i>	

Experiments on Deep Morphological Inflection	382
<i>Akhilesh Sudhakar, Rajesh Kumar Mundotiya, and Anil Kumar Singh</i>	
Identification of Lemmatization Errors Using Neural Models	399
<i>Attila Novák and Borbála Novák</i>	
Full Inflection Learning Using Deep Neural Networks	408
<i>Octavia-Maria Şulea, Liviu P. Dinu, and Bogdan Dumitru</i>	
Semi-supervised Textual Entailment on Indonesian Wikipedia Data	416
<i>Ken Nabila Setya and Rahmad Mahendra</i>	
Author Index	429

Semantics and Text Similarity



Finding Word Sense Embeddings of Known Meaning

Lyndon White, Roberto Togneri, Wei Liu^(✉), and Mohammed Bannamoun

The University of Western Australia, 35 Stirling Highway, Crawley,
Western Australia, Australia

lyndon.white@research.uwa.edu.au,

{roberto.togneri,wei.liu,mohammed.bannamoun}@uwa.edu.au

Abstract. Word sense embeddings are vector representations of polysemous words – words with multiple meanings. These induced sense embeddings, however, do not necessarily correspond to any dictionary senses of the word. To overcome this, we propose a method to find new sense embeddings with known meaning. We term this method refitting, as the new embedding is fitted to model the meaning of a target word in an example sentence. The new lexically refitted embeddings are learnt using the probabilities of the existing induced sense embeddings, as well as their vector values. Our contributions are threefold: (1) The refitting method to find the new sense embeddings; (2) a novel smoothing technique, for use with the refitting method; and (3) a new similarity measure for words in context, defined by using the refitted sense embeddings. We show how our techniques improve the performance of the Adaptive Skip-Gram sense embeddings for word similarity evaluation; and how they allow the embeddings to be used for lexical word sense disambiguation.

Keywords: Word sense embeddings · Polysemous words · Refitting methods

1 Introduction

Popular word embedding vectors, such as Word2Vec, represent a word's semantic meaning and its syntactic role as a point in a vector space [1, 2]. As each word is only given one embedding, such methods are restricted to the representation of only a single combined sense, or meaning, of the word. *Word sense embeddings* generalise word embeddings to handle polysemous and homonymous words. Often these sense embeddings are learnt through unsupervised Word Sense Induction (WSI) [3–6]. The induced sense embeddings are unlikely to directly coincide with any set of human defined meaning at all, i.e. they will not match lexical senses such as those defined in a lexical dictionary, e.g. WordNet [7]. These induced senses may be more specific, more broad, or include the meanings of jargon not in common use.

One may argue that WSI systems can capture better word senses than human lexicographers do manually. However, this does not mean that induced senses

can replace standard lexical senses. It is important to appreciate the vast wealth of existing knowledge defined around lexical senses. Methods to link induced senses to lexical senses allow us to take advantage of both worlds.

We propose a *refitting method* to generate a sense embedding vector that matches with a labelled lexical sense. Given an example sentence with the labelled lexical sense of a particular word, the refitting method algorithmically combines the induced sense embeddings of the target word such that the likelihood of the example sentence is maximised. We find that in doing so, the sense of the word in that sentence is captured. With the refitting, the induced sense embeddings are now able to be used in more general situations where standard senses, or user defined senses are desired.

Refitting word sense vectors to match a lexicographical sense inventory, such as WordNet or a translator’s dictionary, is possible if the sense inventory features at least one example of the target sense’s use. Our method allows this to be done very rapidly, and from only the single example of use this has with possible applications in low-resource languages.

Refitting can also be used to fit to a user provided example, giving a specific sense vector for that use. This has strong applications in information retrieval. The user can provide an example of a use of the word they are interested in. For example, searching for documents about “banks” as in “the river banks were very muddy”. By generating an embedding for that specific sense, and by comparing with the generated embeddings in the indexed documents, we can not only pick up on suitable uses of other-words for example “beach” and “shore”, but also exclude different usages, for example of a financial bank. The method we propose, using our refitted embeddings, has lower time complexity than AvgSimC [3], the current standard method for evaluating the similarity of words in context. This is detailed in Sect. 5.1.

We noted during refitting, that a single induced sense would often dominate the refitted representation. It is rare in natural language for the meaning to be so unequivocal. Generally, a significant overlap exists between the meaning of different lexical senses, and there is often a high level of disagreement when humans are asked to annotate a corpus [8]. We would expect that during refitting there would likewise be contention over the most likely induced sense. Towards this end, we develop a smoothing method, which we call *geometric smoothing* that de-emphasises the sharp decisions made by the (unsmoothed) refitting method. We found that this significantly improves the results. This suggests that the sharpness of sense decisions is an issue with the language model, which smoothing can correct. The geometric smoothing method is presented in Sect. 3.2.

We demonstrate the refitting method on sense embedding vectors induced using Adaptive Skip-Grams (AdaGram) [6], as well as our own simple greedy word sense embeddings. The method is applicable to any skip-gram-like language model that can take a sense vector as its input, and can output the probability of a word appearing in that sense’s context.

The rest of the paper is organised as follows: Sect. 2 briefly discusses two areas of related works. Section 3 presents our refitting method, as well as our

proposed geometric smoothing method. Section 4 describes the WSI embedding models used in the evaluations. Section 5 defines the RefittedSim measure for word similarity in context, and presents its results. Section 6 shows how the refitted sense vectors can be used for lexical WSD. Finally, the paper concludes in Sect. 7.

2 Related Works

2.1 Directly Learning Lexical Sense Embeddings

In this area of research, the induction of word sense embeddings is treated as a supervised, or semi-supervised task, that requires sense labelled corpora for training.

Iacobacci et al. [9] use a Continuous Bag of Word language model [1], using word senses as the labels rather than words. This is a direct application of word embedding techniques. To overcome the lack of a large sense labelled corpus, Iacobacci et al. use a 3rd party WSD tool, BabelFly [10], to add sense annotations to a previously unlabelled corpus.

Chen et al. [11] use a supervised approach to train sense vectors, with an unsupervised WSD labelling step. They partially disambiguate their training corpus, using word sense vectors based on WordNet; and use these labels to train their embeddings. This relabelled data is then used as training data, for finding sense embeddings using skip-grams.

Our refitting method learns a new sense embedding as a weighted sum of existing induced sense embeddings of the target word. Refitting is a one-shot learning solution, as compared to the approaches used in the works discussed above. A notable advantage is the time taken to add a new sense. Adding a new sense is practically instantaneous, and replacing the entire sense inventory, of several hundred thousand senses, is only a matter of a few hours. Whereas for the existing approaches this would require repeating the training process, which will often take several days. Refitting is a process done to word sense embeddings, rather than a method for finding sense embeddings from a large corpus.

2.2 Mapping Induced Senses to Lexical Senses

By defining a stochastic map between the induced and lexical senses, Agirre et al. [12], propose a general method for allowing WSI systems to be used for WSD. Their work was used in SemEval-2007 Task 02 [13] to evaluate all entries. Agirre et al. use a mapping corpus to find the probability of a lexical sense, given the induced sense according to the WSI system. This is more general than the approach we propose here, which only works for sense embedding based WSI. By exploiting the particular properties of sense embedding based WSI systems we propose a system that can better facilitate the use of this subset of WSI systems for WSD.

3 Proposed Refitting Framework

The key contribution of this work is to provide a way to synthesise a word sense embedding given only a single example sentence and a set of pretrained sense embedding vectors. We termed this *refitting* the sense vectors. By refitting the unsupervised vectors we define a new vector, that lines up with the specific meaning of the word from the example sentence.

This can be looked at as a one-shot learning problem, analogous to regression. The training of the induced sense, and of the language model, can be considered an unsupervised pre-training step. The new word sense embedding should give a high value for the likelihood of the example sentence, according to the language model. It should also generalise to give a high likelihood of other contexts where this word sense occurs.

We initially attempted to directly optimise the sense vector to predict the example. We applied the L-BFGS [14] optimisation algorithm with the sense vector being the parameter being optimised over, and the objective being to maximise the probability of the example sentence according to the language model. This was found to generalise poorly, due to over-fitting, and to be very slow. Rather than a direct approach, we instead take inspiration from the locally linear relationship between meaning and vector position that has been demonstrated for word embeddings [1, 15, 16].

To refit the induced sense embeddings to a particular meaning of a word, we express that a new embedding as a weighted combination of the induced sense vectors. The weight is determined by the probability of each induced sense given the context.

Given a collection of induced (unlabelled) embeddings $\mathbf{u} = u_1, \dots, u_{n_u}$, and an example sentence $\mathbf{c} = w_1, \dots, w_{n_c}$ we define a function $l(\mathbf{u} | \mathbf{c})$ which determines the refitted sense vector, from the unsupervised vectors and the context as:

$$l(\mathbf{u} | \mathbf{c}) = \sum_{\forall u_i \in \mathbf{u}} u_i P(u_i | \mathbf{c}) \quad (1)$$

Bayes' Theorem can be used to estimate the posterior predictive distribution $P(u_i | \mathbf{c})$.

Bengio et al. [17] describe a similar method to Eq. (1) for finding (single sense) word embeddings for words not found in their vocabulary. The formula they give is as per Eq. (1), but summing over the entire vocabulary of words (rather than just \mathbf{u}).

3.1 A General WSD Method

Using the language model and application of Bayes' theorem, we define a general word sense disambiguation method that can be used for refitting (Eq. (1)), and for lexical word sense disambiguation (see Sect. 6). This is a standard approach of using Bayes' theorem [5, 6]. We present it here for completeness.

The context is used to determine which sense is the most suitable for this use of the *target word* (the word being disambiguated). Let $\mathbf{s} = (s_1, \dots, s_n)$, be the collection of senses for the target word¹.

Let $\mathbf{c} = (w_1, \dots, w_{n_c})$ be a sequence of words making up the context of the target word. For example for the target word *kid*, the context could be $\mathbf{c} = (\textit{wow the wool from the, is, so, soft, and, fluffy})$, where *kid* is the central word taken from between *the* and *fluffy*.

For any particular sense, s_i , the multiple sense skip-gram language model can be used to find the probability of a word w_j occurring in the context: $P(w_j | s_i)$. By assuming the conditional independence of each word w_j in the context, given the sense embedding s_i , the probability of the context can be calculated:

$$P(\mathbf{c} | s_i) = \prod_{\forall w_j \in \mathbf{c}} P(w_j | s_i) \quad (2)$$

The correctness of the conditional independence assumption depends on the quality of the representation – the ideal sense representation would fully capture all information about the contexts it can appear in – thus the other contexts elements would not present any additional information, and so $P(w_a | w_b, s_i) = P(w_a | s_i)$. Given this, we have an estimate of $P(\mathbf{c} | s_i)$ which can be used to find $P(s_i | \mathbf{c})$. However, a false assumption of independence contributes towards overly sharp estimates of the posterior distribution [18], which we seek to address in Sect. 3.2 with geometric smoothing.

Bayes’ Theorem is applied to this context likelihood function $P(\mathbf{c} | s_i)$ and a prior for the sense $P(s_i)$ to allow the posterior probability to be found:

$$P(s_i | \mathbf{c}) = \frac{P(\mathbf{c} | s_i)P(s_i)}{\sum_{s_j \in \mathbf{s}} P(\mathbf{c} | s_j)P(s_j)} \quad (3)$$

This is the probability of the sense given the context.

3.2 Geometric Smoothing for General WSD

During refitting, we note that often one induced sense would be calculated as having much higher probability of occurring than the others (according to Eq. 3). This level of certainty is not expected to occur in natural languages, ambiguity is almost always possible. To resolve such dominance problems, we propose a new *geometric smoothing* method. This is suitable for smoothing posterior probability estimates derived from products of conditionally independent likelihoods. It smooths the resulting distribution, by shifting all probabilities to be closer to the uniform distribution.

We hypothesize that the sharpness of probability estimates from Eq. (3) is a result of data sparsity, and of a false independence assumption in Eq. (2). This

¹ As this part of our method is used with both the unsupervised senses and the lexical senses, referred to as \mathbf{u} and \mathbf{l} respectively in other parts of the paper, here we use a general sense \mathbf{s} to avoid confusion.

is well known to occur for n-gram language models [18]. Word-embeddings language models largely overcome the data sparsity problem due to weight sharing effects [17]. We suggest that the problem remains for word sense embeddings, where there are many more classes. Thus the training data must be split further between each sense than it was when split for each word. The power law distribution of word use [19] is compounded by word senses within those used also following the a power law distribution [20]. Rare senses are liable to over-fit to the few contexts they do occur in, and so give disproportionately high likelihoods to contexts that those are similar to. We propose to handle these issues through additional smoothing.

We consider replacing the unnormalised posterior with its n_c -th root, where n_c is the length of the context. We replace the likelihood of Eq. (2) with $P_S(\mathbf{c} \mid s_i) = \prod_{w_j \in \mathbf{c}} \sqrt[n_c]{P(w_j \mid s_i)}$. Similarly, we replace the prior with: $P_S(s_i) = \sqrt[n_c]{P(w_j \mid s_i)}$ When this is substituted into Eq. (3), it becomes a smoothed version of $P(s_i \mid \mathbf{c})$.

$$P_S(s_i \mid \mathbf{c}) = \frac{\sqrt[n_c]{P(\mathbf{c} \mid s_i)P(s_i)}}{\sum_{s_j \in \mathbf{s}} \sqrt[n_c]{P(\mathbf{c} \mid s_j)P(s_j)}} \quad (4)$$

The motivation for taking the n_c -th root comes from considering the case of the uniform prior. In this case $P_S(\mathbf{c} \mid s_i)$ is the geometric mean of the individual word probabilities $P_S(w_j \mid s_i)$. Consider, if one has two context sentences, $\mathbf{c} = \{w_1, \dots, w_{n_c}\}$ and $\mathbf{c}' = \{w'_1, \dots, w'_{n_{c'}}\}$, such that $n_{c'} > n_c$ then using Eq. (2) to calculate $P(\mathbf{c} \mid s_i)$ and $P(\mathbf{c}' \mid s_i)$ will result in incomparable results as additional number of probability terms will dominate – often significantly more than the relative values of the probabilities themselves. The number of words that can occur in the context of any given sense is very large – a large portion of the vocabulary. We would expect, averaging across all words, that each addition word in the context would decrease the probability by a factor of $\frac{1}{V}$, where V is the vocabulary size. The expected probabilities for $P(\mathbf{c} \mid s_i)$ is $\frac{1}{V^{n_c}}$ and for $P(\mathbf{c}' \mid s_i)$ is $\frac{1}{V^{n_{c'}}$. As $n_{c'} > n_c$, thus we expect $P(\mathbf{c}' \mid s_i) \ll P(\mathbf{c} \mid s_i)$. Taking the n_c -th and $n_{c'}$ -th roots of $P(\mathbf{c} \mid s_i)$ and $P(\mathbf{c}' \mid s_i)$ normalises these probabilities so that they have the same expected value; thus making a context-length independent comparison possible. When this normalisation is applied to Eq. (3), we get the smoothing effect.

4 Experimental Sense Embedding Models

We trained two sense embedding models, AdaGram [6] and our own Greedy Sense Embedding method. During training we use the Wikipedia dataset as used by Huang et al. [4]. However, we do not perform the extensive preprocessing used in that work.

Most of our evaluations are carried out on Adaptive SkipGrams (AdaGram) [6]. AdaGram is a non-parametric Bayesian extension of Skip-gram. It learns a number of different word senses, as are required to properly model the language.

We use the implementation² provided by the authors with minor adjustments for Julia [21] v0.5 compatibility.

The AdaGram model was configured to have up to 30 senses per word, where each sense is represented by a 100 dimension vector. The sense threshold was set to 10^{-10} to encourage many senses. Only words with at least 20 occurrences are kept, this gives a total vocabulary size of 497,537 words.

To confirm that our techniques are not merely a quirk of the AdaGram method or its implementation, we implemented a new simple baseline word sense embedding method. This method starts with a fixed number of randomly initialised embeddings, then greedily assigns each training case to the sense which predicts it with the highest probability (using Eq. (3)). The task remains the same: using skip-grams with hierarchical softmax to predict the context words for the input word sense. This is similar to [22], however it is using collocation probability, rather than distance in vector-space as the sense assignment measure. Our implementation is based on a heavily modified version of Word2Vec.jl³.

This method is intrinsically worse than AdaGram. Nothing in the model encourages diversification and specialisation of the embeddings. Manual inspection reveals that a variety of senses are captured, though with significant repetition of common senses, and with rare senses being missed. Regardless of its low quality, it is a fully independent method from AdaGram, and so is suitable for our use in checking the generalisation of the refitting techniques.

The vocabulary used is smaller than for the AdaGram model. Words with at least 20,000 occurrences are allocated 20 senses. Words with at least 250 occurrences are restricted to a single sense. The remaining rare words are discarded. This results in a vocabulary size of 88,262, with 2,796 words having multiple senses. We always use a uniform prior, as the model does not facilitate easy calculation of the prior.

5 Similarity of Words in Context

Estimating word similarity with context is the task of determining how similar words are, when presented with the context they occur in. The goal of this task is to match human judgements of word similarity. For each of the target words and contexts; we use refitting on the target word to create a word sense embedding specialised for the meaning in the context provided. Then the similarity of the refitted vectors can be measured using cosine distance (or similar). By measuring similarity this way, we are defining a new similarity measure.

Reisinger and Mooney [3] define a number of measures for word similarity suitable for use with sense embeddings. The most successful was AvgSimC, which has become the gold standard method for use on similarity tasks. It has been used with great success in many works [4, 5, 11].

² <https://github.com/sbos/AdaGram.jl>.

³ <https://github.com/tanmaykm/Word2Vec.jl/>.

AvgSimC is defined using distance metric d (normally cosine distance) as:

$$\text{AvgSimC}((\mathbf{u}, \mathbf{c}), (\mathbf{u}', \mathbf{c}')) = \frac{1}{n \times n'} \sum_{u_i \in \mathbf{u}} \sum_{u'_j \in \mathbf{u}'} P(u_i | \mathbf{c}) P(u'_j | \mathbf{c}') d(u_i, u'_j) \quad (5)$$

for contexts \mathbf{c} and \mathbf{c}' , the contexts of the two words to be compared, and for $\mathbf{u} = \{u_1, \dots, u_n\}$ and $\mathbf{u}' = \{u'_1, \dots, u'_{n'}\}$ the respective sets of induced senses of the two words.

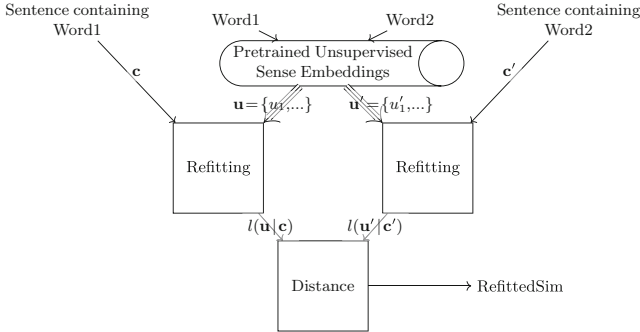


Fig. 1. Block diagram for RefittedSim similarity measure

5.1 A New Similarity Measure: RefittedSim

We define a new similarity measure, RefittedSim, as the distance between the refitted sense embeddings. As shown in Fig. 1 the example contexts are used to refit the induced sense embeddings of each word. This is a direct application of Eq. (1).

Using the same definitions as in Eq. (5), RefittedSim is defined as:

$$\text{RefittedSim}((\mathbf{u}, \mathbf{c}), (\mathbf{u}', \mathbf{c}')) = d(l(\mathbf{u} | \mathbf{c}), l(\mathbf{u}' | \mathbf{c}')) = d\left(\sum_{u_i \in \mathbf{u}} u_i P(u_i | \mathbf{c}), \sum_{u'_j \in \mathbf{u}'} u'_j P(u'_j | \mathbf{c}')\right) \quad (6)$$

AvgSimC is a probability weighted average of pairwise computed distances for each sense vector. Whereas RefittedSim is a single distance measured between the two refitted vectors – which are the probability weighted averages of the original unsupervised sense vectors.

There is a notable difference in time complexity between AvgSimC and RefittedSim. AvgSimC has time complexity $O(n \|\mathbf{c}\| + n' \|\mathbf{c}'\| + n \times n')$, while RefittedSim has $O(n \|\mathbf{c}\| + n' \|\mathbf{c}'\|)$. The product of the number of senses of each word $n \times n'$, may be small for dictionary senses, but it is often large for induced senses. Dictionaries tend to define only a few senses per word – the average⁴

⁴ It should be noted, though, that the number of meanings is not normally distributed [23].

number of senses per word in WordNet is less than three [7]. For induced senses, however, it is often desirable to train many more senses, to get better results using the more fine-grained information. Reisinger and Mooney [3] found optimal results in several evaluations near 50 senses. In such cases the $O(n \times n')$ is significant, avoiding it with RefittedSim makes the similarity measure more useful for information retrieval.

5.2 Experimental Setup

We evaluate our refitting method using Stanford’s Contextual Word Similarities (SCWS) dataset [4]. During evaluation, each context paragraph is limited to 5 words to either side of the target word, as in the training.

Table 1. Spearman rank correlation $\rho \times 100$ when evaluated on the SCWS task.

(a) For varying hyper-parameters.					(b) Compared to other methods RefittedSim-S is with smoothing, and RefittedSim-SU is with uniform prior			
Method	Geometric Smoothing	Use Prior	AvgSimC	RefittedSim	Paper	Embedding	Similarity	$\rho \times 100$
AdaGram	T	T	53.8	64.8	This paper	AdaGram	AvgSimC	43.8
AdaGram	T	F	36.1	65.0	This paper	AdaGram	RefittedSim-S	64.8
AdaGram	F	T	43.8	47.8	This paper	AdaGram	RefittedSim-SU	65.0
AdaGram	F	F	20.7	24.1	[4]	Huang et al.	AvgSimC	65.7
Greedy	T	F	23.6	49.7	[4]	Pruned tf-idf	AvgSimC	60.5
Greedy	F	F	22.2	40.7	[11]	Chen et al.	AvgSimC	68.9
					[5]	Tian et al.	AvgSimC	65.4
					[5]	Tian et al.	MaxSim	65.6
					[9]	SenseEmbed	Min Tanimoto	58.9
					[9]	SenseEmbed	Weighted Tanimoto	62.4

5.3 Results

Table 1a shows the results of our evaluations on the SCWS similarity task. A significant improvement can be seen by applying our techniques.

The RefittedSim method consistently outperforms AvgSimC across all configurations. Similarly geometric smoothing consistently improves performance both for AvgSimC and for RefittedSim. The improvement is significantly more for RefittedSim than for AvgSimC results. In general using the unsupervised sense prior estimate from the AdaGram model, improves performance – particularly for AvgSimC. The exception to this is with RefittedSim with smoothing, where it makes very little difference. Unsurprisingly, given its low quality, the Greedy embeddings are always outperformed by AdaGram. It is not clear if these improvements will transfer to clustering based methods due to the differences in how the sense probability is estimated, compared to the language model based method evaluated on in Table 1a.

Table 1b compares our results with those reported in the literature using other methods. These results are not directly comparable, as each method uses

a different training corpus, with different preprocessing steps, which can have significant effects on performance. It can be seen that by applying our techniques we bring the results of our AdaGram model from very poor ($\rho \times 100 = 43.8$) when using normal AvgSimC without smoothing, up to being competitive with other models, when using RefittedSim with smoothing. The method of Chen et al. [11], has a significant lead on the other results presented. This can be attributed to its very effective semi-supervised fine-tuning method. This suggests a possible avenue for future development in using refitted sense vectors to relabel a corpus, and then performing fine-tuning similar to that done by Chen et al.

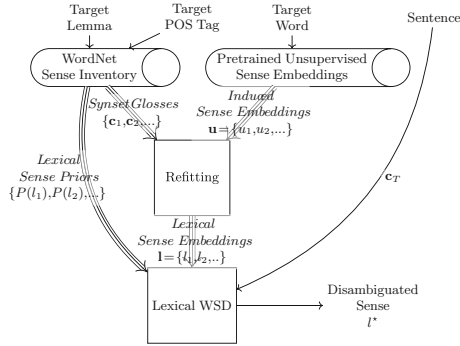


Fig. 2. Block diagram for performing WSD using refitting.

6 Word Sense Disambiguation

6.1 Refitting for Word Sense Disambiguation

Once refitting has been used to create sense vectors for lexical word senses, an obvious use of them is to perform word sense disambiguation. In this section we refer to the lexical word sense disambiguation problem, i.e. to take a word and find its dictionary sense; whereas the methods discussed in Eqs. (3) and (4) consider the more general problem, as applicable to disambiguating lexical or induced word senses depending on the inputs. Our overall process shown in Fig. 2 uses both: first disambiguating the induced senses as part of refitting, then using the refitted sense vectors to find the most likely dictionary sense.

First, refitting is used to transform the induced sense vectors into lexical sense vectors. We use the targeted word’s lemma (i.e. base form), and part of speech (POS) tag to retrieve all possible definitions of the word (Glosses) from WordNet; there is one gloss per sense. These glosses are used as the example sentence to perform refitting (see Sect. 3). We find embeddings, $\mathbf{l} = \{l_1, \dots, l_{n_l}\}$ for each of the lexical word senses using Eq. (1). These lexical word senses are still supported by the language model, which means one can apply the general

WSD method to determine the posterior probability of a word sense, given an observed context.

When given a sentence \mathbf{c}_T , containing a target word to be disambiguated, the probability of each lexical word sense $P(l_i | \mathbf{c}_T)$, can be found using Eq. (3) (or the smoothed version Eq. (4)), over the lexically refitted sense embeddings. Then, selecting the correct sense is simply selecting the most likely sense:

$$l^*(\mathbf{l}, \mathbf{c}_T) = \underset{\forall l_i \in \mathbf{l}}{\operatorname{argmax}}: P(l_i | \mathbf{c}_T) = \underset{\forall l_i \in \mathbf{l}}{\operatorname{argmax}}: \frac{P(\mathbf{c}_T | l_i)P(l_i)}{\sum_{\forall l_j \in \mathbf{l}} P(\mathbf{c}_T | l_j)P(l_j)} \quad (7)$$

6.2 Lexical Sense Prior

WordNet includes frequency counts for each word sense based on Semcor [24]. These form a prior for $P(l_i)$. The comparatively small size of Semcor means that many word senses do not occur at all. We apply add-one smoothing to remove any zero counts. This is in addition to using our proposed geometric smoothing as an optional part of the general WSD. Geometric smoothing serves a different (but related) purpose, of decreasing the sharpness of the likelihood function – not of removing impossibilities from the prior.

6.3 Experimental Setup

The WSD performance is evaluated on the SemEval 2007 Task 7.

We use the weighted mapping method of Agirre et al. [12], (see Sect. 2.2) as a baseline alternative method for using WSI senses for WSD. We use Semcor as the mapping corpus, to derive the mapping weights.

The second baseline we use is the Most Frequent Sense (MFS). This method always disambiguates any word as having its most common meaning. Due to the power law distribution of word senses, this is a very effective heuristic [20]. We also consider the results when using a backoff to MSF when a method is unable to determine the word sense the method can report the MFS instead of returning no result (a non-attempt).

6.4 Word Sense Disambiguation Results

The results of employing our method for WSD, are shown in Table 2. Our results using smoothed refitting, both with AdaGram and Greed Embeddings with back-off, outperform the MSF baseline [25] – noted as a surprisingly hard baseline to beat [11].

The mapping method [12] was not up to the task of mapping unsupervised senses to supervised senses, on this large scale task. The Refitting method works better. Though refitting is only usable for language-model embedding WSI, the mapping method is suitable for all WSI systems.

While not directly comparable due to the difference in training data, we note that our Refitted results, are similar in performance, as measured by F1

Table 2. Results on SemEval 2007 Task 7 – course-all-words disambiguation. The *-S* marks results using geometric smoothing. The * marks results with MSF backoff.

Method	Attempted	Precision	Recall	F1
Refitted-S AdaGram	99.91%	0.799	0.799	0.799
Refitted AdaGram	99.91%	0.774	0.773	0.774
Refitted-S Greedy	79.95%	0.797	0.637	0.708
Refitted-S Greedy*	100.00%	0.793	0.793	0.793
Refitted Greedy	79.95%	0.725	0.580	0.645
Refitted Greedy*	100.00%	0.793	0.793	0.793
Mapped AdaGram	84.31%	0.776	0.654	0.710
Mapped AdaGram*	100.00%	0.736	0.736	0.736
MFS baseline	100.00%	0.789	0.789	0.789

score, to the results reported by Chen et al. [11]. AdaGram with smoothing, and Greedy embeddings with backoff have close to the same result as reported for L2R with backoff – with the AdaGram slightly better and the Greedy embeddings slightly worse. They are exceeded by the best method reported in that paper: S2C method with backoff. Comparison to non-embedding based methods is not discussed here for brevity. Historically state of the art systems have functioned very differently; normally by approaching the WSD task by more direct means.

Our results are not strong enough for Refitted AdaGram to be used as a WSD method on its own, but do demonstrate that the senses found by refitting are capturing the information from lexical senses. It is now evident that the refitted sense embeddings are able to perform WSD, which was not possible with the unsupervised senses.

7 Conclusion

A new method is proposed for taking unsupervised word embeddings, and adapting them to align to particular given lexical senses, or user provided usage examples. This refitting method thus allows us to find word sense embeddings with known meaning. This method can be seen as a one-shot learning task, where only a single labelled example of each class is available for training. We show how our method can be used to create embeddings to evaluate the similarity of words, given their contexts.

This allows us to propose a new similarity measuring method, Refitted-Sim. The performance of RefittedSim on AdaGram is comparable to the results reported by the researchers of other sense embeddings techniques using AvgSimC, but its time complexity is significantly lower. We also demonstrate how similar refitting principles can be used to create a set of vectors that are aligned to the meanings in a sense inventory, such as WordNet.

We show how this can be used for word sense disambiguation. On this difficult task, it performs marginally better than the hard to beat MFS baseline, and significantly better than a general mapping method used for working with WSI senses on lexical WSD tasks. As part of our method for refitting, we present a geometric smoothing to overcome the issues of overly dominant senses probability estimates. We show that this significantly improves the performance. Our refitting method provides effective bridging between the vector space representation of meaning, and the traditional discrete lexical representation. More generally it allows a sense embedding to be created to model the meaning of a word in any given sentence. Significant applications of sense embeddings in tasks such as more accurate information retrieval thus become possible.

References

1. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. [arXiv:1301.3781](https://arxiv.org/abs/1301.3781) (2013)
2. Pennington, J., Socher, R., Manning, C.D.: Glove: global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014), pp. 1532–1543 (2014)
3. Reisinger, J., Mooney, R.J.: Multi-prototype vector-space models of word meaning. In: Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, pp. 109–117. Association for Computational Linguistics (2010)
4. Huang, E.H., Socher, R., Manning, C.D., Ng, A.Y.: Improving word representations via global context and multiple word prototypes. In: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Vol. 1, pp. 873–882. Association for Computational Linguistics (2012)
5. Tian, F., et al.: A probabilistic model for learning multi-prototype word embeddings. In: COLING, pp. 151–160 (2014)
6. Bartunov, S., Kondrashkin, D., Osokin, A., Vetrov, D.P.: Breaking sticks and ambiguities with adaptive skip-gram. CoRR abs/1502.07257 (2015)
7. Miller, G.A.: Wordnet: a lexical database for English. *Commun. ACM* **38**, 39–41 (1995)
8. Véronis, J.: A study of polysemy judgements and inter-annotator agreement. In: Programme and Advanced Papers of the Senseval workshop, pp. 2–4 (1998)
9. Iacobacci, I., Pilehvar, M.T., Navigli, R.: Sensemed: learning sense embeddings for word and relational similarity. In: Proceedings of ACL, pp. 95–105 (2015)
10. Moro, A., Raganato, A., Navigli, R.: Entity linking meets word sense disambiguation: a unified approach. *Trans. Assoc. Comput. Linguist.* **2**, 231–244 (2014)
11. Chen, X., Liu, Z., Sun, M.: A unified model for word sense representation and disambiguation. In: EMNLP, pp. 1025–1035. Citeseer (2014)
12. Agirre, E., Martínez, D., De Lacalle, O.L., Soroa, A.: Evaluating and optimizing the parameters of an unsupervised graph-based WSD algorithm. In: Proceedings of the First Workshop on Graph Based Methods for Natural Language Processing, Association for Computational Linguistics, pp. 89–96 (2006)
13. Agirre, E., Soroa, A.: Semeval-2007 task 02: evaluating word sense induction and discrimination systems. In: Proceedings of the 4th International Workshop on Semantic Evaluations. SemEval 2007, Stroudsburg, PA, USA, pp. 7–12. Association for Computational Linguistics (2007)

14. Nocedal, J.: Updating quasi-newton matrices with limited storage. *Math. Comput.* **35**, 773–782 (1980)
15. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: *Advances in Neural Information Processing Systems*, pp. 3111–3119 (2013)
16. Mikolov, T., Yih, W.t., Zweig, G.: Linguistic regularities in continuous space word representations. In: *HLT-NAACL*, pp. 746–751 (2013)
17. Bengio, Y., Ducharme, R., Vincent, P., Janvin, C.: A neural probabilistic language model. *J. Mach. Learn. Res.* **3**, 137–186 (2003)
18. Rosenfeld, R.: Two decades of statistical language modeling: where do we go from here? *Proc. IEEE* **88**, 1270–1278 (2000)
19. Zipf, G.: *Human Behavior and the Principle of Least Effort: An Introduction to Human Ecology*. Addison-Wesley Press, Cambridge (1949)
20. Kilgariff, A.: How dominant is the commonest sense of a word? In: Sojka, P., Kopeček, I., Pala, K. (eds.) *TSD 2004. LNCS (LNAI)*, vol. 3206, pp. 103–111. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-30120-2_14
21. Bezanson, J., Edelman, A., Karpinski, S., Shah, V.B.: Julia: a fresh approach to numerical computing. *SIAM Rev.* **59**, 65–98 (2014)
22. Neelakantan, A., Shankar, J., Passos, A., McCallum, A.: Efficient non-parametric estimation of multiple embeddings per word in vector space. *arXiv preprint arXiv:1504.06654* (2015)
23. Zipf, G.K.: The meaning-frequency relationship of words. *J. Gen. Psychol.* **33**, 251–256 (1945)
24. Tengi, R.I.: Design and implementation of the WordNet lexical database and searching software. In: *WordNet: An Electronic Lexical Database*, p. 105. The MIT Press, Cambridge (1998)
25. Navigli, R., Litkowski, K.C., Hargraves, O.: Semeval-2007 task 07: coarse-grained English all-words task. In: *Proceedings of the 4th International Workshop on Semantic Evaluations. SemEval 2007*, Stroudsburg, PA, USA, Association for Computational Linguistics, pp. 30–35 (2007)



Using Shallow Semantic Analysis to Implement Automated Quality Assessment of Web Health Care Information

Yanjun Zhang¹, Robert E. Mercer¹ , Jacquelyn Burkell¹ ,
and Hong Cui² 

¹ The University of Western Ontario, London, ON, Canada
mercer@csd.uwo.ca

² University of Arizona, Tucson, AZ, USA

Abstract. Evidence-based clinical practice guidelines have been widely used as an objective rating instrument for assessing the content quality of health care information on the web. In many previous studies, human raters check the concordance between text content and evidence-based practice guidelines in order to evaluate information accuracy and completeness. However, human rating cannot be a practical solution, particularly when there is an extremely large volume of health care information on the web. This study explores a semantics-based approach to identify health care information content in web documents with reference to evidence-based health care guidelines. With this approach terms and phrases in English are extracted and transformed into semantic concepts and units. Thus, web text is transformed, sentence by sentence, into a semantic representation which computer programs can classify depending on whether the content of a sentence is in concordance with evidence-based guidelines or not. Through aggregating the classification result of all sentences in a web document, computer programs are able to generate for each document a quality score indicating the number of unique evidence-based guidelines that are referred to in the document. In a test using a set of depression treatment web pages and evidence-based clinical guidelines, the quality rating performance of the computer system is shown to be close to human quality rating performance.

Keywords: Semantics-based quality rating approach · Natural language processing · Shallow semantic analysis · Evidence-based clinical practice · Semantics-based classification · Web health care information

1 Introduction

The past decade has witnessed a dramatic expansion in the amount of publicly available health care information on the Web. On the one hand, the demand

for web health care information is huge and keeps growing [14, 16, 17]. On the other hand, the information quality is of extreme variability [5, 7, 12]. In spite of this, different previous studies identified that users are ready to accept health care information from unfamiliar websites and they do not like the burden to verify information quality [15]. Such situations can cause threat to public health, including life-threatening cases [4, 11]. Because of the potential harm that may be caused by inaccurate information, quality assessment of health care information on the web stays a common interest of various health care information stakeholders, including e-health policy makers, information providers/consumers, and information search service providers.

In the last ten years and even longer, researchers have made a lot of efforts and progress in the quality evaluation area, including exploration on establishing quality rating criteria, creating rating tools, etc. One type of commonly used rating criteria is evidence-based health care practice guidelines. As summarized in a systematic review [5] based on 79 distinct quality evaluation studies, health care guidelines are widely utilized by researchers for rating content quality in terms of information accuracy and comprehensiveness. Evaluating such type of quality is just the focus of this study. Our goal is to develop an automated approach to evaluate the information accuracy and comprehensiveness of health care web pages. Other types of quality, such as web site/page design aesthetics, web page readability, etc., referred as presentation quality in [5] are not covered in this study. Depression treatment is the selected subject domain.

2 Related Work

Based on summarization from 79 distinct quality evaluation studies, [5] defined accuracy as the degree of concordance between the web content and generally accepted health care practice. The completeness is also called “comprehensiveness” or “coverage”. Most researchers calculated the proportion of pre-defined clinical guidelines covered by a web source. In practice, the evaluation of accuracy and comprehensiveness requires human raters to read and understand web page content in order to compare web content against guidelines. The objectiveness and effectiveness of guidelines are advantages compared with some other criteria such as accountability meta-information of web sites or pages [2]. However, the use of health care guidelines for rating health care web information quality is limited by the dependency on human efforts. Due to the explosion of health care information on the web, it would be an impractical practice to have human raters manually evaluate all related web pages through reading content sentence by sentence.

On the other hand, many previous studies have investigated the association between content quality and accountability of web sites or pages, and attempted to use the latter as quality indicators since they are subject independent. Investigated indicators range from bibliographic metadata (e.g., authorship, editorial board, references) from the print world (e.g., [1, 3, 18, 19]) to web-unique features (e.g., [6, 7]) such as site domain name suffix, hyperlinks received from external web sites, Google PageRank, etc., and to quality seal such as HONcode certificate [9]. Specifically, Wang and Liu (2007) [20] developed an Automatic Indicator

Detection Tool to collect indirect quality indicators using an HTML parser. In their testing, the performance of detecting such indicators reached 93% recall and 98% precision. However, their study did not include quality evaluation test to prove that detected indicators can accurately predicate content quality. In fact, different research groups [6, 8, 10, 13] found in their studies that the association between these indicators and the content quality of web health care information is inconsistent in different health care subjects, putting the validity and reliability of these non-content based indicators in question.

In comparison, rating the quality directly based on the web site/page content is relatively more reliable than relying on metadata indicators. To the best of our knowledge, Griffiths et al. (2005) [8] did a first attempt of automated quality rating based on processing the web text content. In their study, they used evidence-based depression treatment guidelines (CEBMH 1998) as rating criteria, and human rating quality scores according to guidelines are standards for evaluating automated rating performance. Their approach is based on information retrieval techniques. They tried to use web pages from training web sites to establish two standard queries, comprising of 20 keywords and 20 two-word phrases with high discriminative power in terms of content quality or relevance respectively. For given web sites, the similarity between the web content and the standard queries are used to calculate the site quality score. In their testing, the automatically rated website scores and the evidence-based human rated scores have strong Pearson correlation equal to 0.85.

3 Method

Griffiths et al. (2005) [8] successfully used the keywords among health care web pages to predicate the information quality. In our study, we try to utilize the semantics of web content, and specifically analyze semantics at sentence level in order to predicate whether a sentence present meanings in concordance with given health care guidelines. Two key issues are explored and solved in order to reach this goal:

1. First, how to create an effective representation of the web text semantics? To compare the web content against evidence based health care guidelines, it is important that our automated quality rating approach can capture and represent text semantics at appropriate extent.
2. Second, with captured semantics, how can computer successfully identify the presentation of health care guidelines in web document?

Semantics-based quality rating approach: overall, this automated rating approach use computer programs to read in every sentence in a web page and check if its content agrees with any pre-defined health care guideline. The quality score is the number of unique guidelines that are referred to in the web page.

Text classification is applied to a single sentence to determine if it presents one or more guidelines. Each guideline has a binary classifier to categorize sentences into “positive” (i.e., a match with guideline) or “negative” group. Features

used to implement text classification mainly include text semantics and relevant metadata. Shallow semantic analysis is used to capture these features and map a sentence into a semantic tag instance. Thus, text classification is done based on semantic tags rather than the original text.

Shallow Semantic Analysis: as the purpose is to convert sentences into semantic tag instances, shallow semantic analysis in this study focuses only on annotating semantically essential units in a sentence. Certainly, every depression treatment guideline has a unique theme. We assume that it contains a specific set of semantic concepts although they could have different variations when expressed in natural language.

First, health care concepts such as health conditions, treatment etc. are one type of the very important semantic units. They are usually nouns or noun phrases. In addition, verbs, adjectives and adverbs are important for describing the relations between semantic concepts. We want to map these elements from text to semantic tags, and also get their part of speech, and their distance between each other. We consider distance as a useful feature because semantic unit pairs which have tight relations likely occur close to each other.

To capture the semantic concepts and above features, we used the Unified Medical Language System (UMLS 2009) to develop our shallow semantic tagging application. UMLS is a free resource provided by the National Library of Medicine in the United States. It provides a knowledge source called Metathesaurus, which include more than 60 controlled vocabularies in biomedical domains such as MeSH, SNOMED, etc.

UMLS also provides supporting software tools to facilitate access and use of UMLS data. Two very useful software packages are used in this study: 1) MetaMap API is used to discover Metathesaurus concepts referred by text. The benefit is that health care terms and variants of a same semantic concept can be unified. For example, text strings “depression”, “Depressive episode” and “depressive illness” are all labeled into MMTx tag “Depressive disorder”. 2) SPECIALIST NLP Tools (including LVG and TaggerClient) facilitate natural language processing by dealing with lexical variation and text analysis tasks in the biomedical domain. Supported NLP functions include sentence splitting, tokenization, POS tagging, lemmatization, etc. We used this tool to transform and filter lexical variants from the original text of sentences. For example, “ceases”, “ceased”, “stopping”, and “stops” can be transformed to a LVG tag “stop”.

We developed JAVA programs to utilize above resources and to use the UMLS tagging results to generate semantic tag instance for sentences. The definition of a semantic tag instance includes header and body parts (see Fig. 1). The instance header includes sentence sequence number in a page, begin and end offset of this sentence, and the original text. In the instance body, the labeled semantic units are saved in sequence. Each labeled unit is either a LVG tag by default or an MMTx tag for Metathesaurus concepts, enclosed by a pair of square brackets and separated by pipelines.

Figure 2 shows two examples of semantic tagging result. A semantic tag instance contains tags for semantically essential words or phrases in a sentence. In the first example, phrase “depressive illness” is converted to a unified concept “Depressive Disorder”. In the second, “tricyclics” was converted to concept

Syntax of Semantic Tag Instance (for a sentence):

```
|===sentence===|sentenceSequenceNum|begin-offset|end-offset|##%#original text of
sentence##% [LVG or MMTx tag]||LVG or MMTx tag||. . .
```

LVG|MMTx Tag Syntax:

```
[tag, begin-offset, end-offset, Term Sequence Num within the hosting sentence,
POS|Semantic Type, (Semantic mapping score)]
```

Fig. 1. Definition of semantic tag instance

1. People with a depressive illness cannot merely "pull themselves together" and get better.

```
|===Sentence===|8|520|610|##%#People with a depressive illness cannot
merely "pull themselves together" and get better. ##%[Per-
sons,520,525,0,Population Group,1000] [Depressive disorder, 534,543,3, Mental
or Behavioral Dysfunction,1000]||merely,560, 565,6, adv||[pull,569,572,7,adv] |[to-
gether,585,592,9,adv]||get,599,601,11,verb||[best,603,608,12,adj]
```

2. Side effects of tricyclics, which vary from person to person, may include dry mouth, blurred vision, constipation, problems passing urine, sweating, light-headedness and excessive drowsiness.

```
|===Sentence===|3|365|557|##%#Side effects of tricyclics, which vary from per-
son to person, may include dry mouth, blurred vision, constipation, prob-
lems passing urine, sweating, light-headedness and excessive drowsiness.##%[effect
side,365,376,0,noun] |[Antidepressive Agents,381,390,2,Organic Chemical, Phar-
macologic substance,1000]||vary,399, 402,4,verb] |[Persons,409, 414,6,Population
Group,1000]}. . . |[excessive,535,543,20,adj]||[Drowsiness,545, 554,21,Sign or Symp-
tom,861]
```

Fig. 2. Example of semantic tag instances

“Antidepressive Agents”. In addition, each tag also contains the position meta-data, POS metadata, etc. For example, “effect side” is the semantic tag for text “side effects” which has position index from 365 to 376. The term index of this semantic unit in this sentence is 0. The part of speech is NOUN.

Rule-based Classification: The generated semantic tag instances are the input for text classification. Each guideline has a dedicated classifier to make binary prediction regarding whether a given sentence agrees with the specific guideline. This study used a rule-based classification to solve the problem. For each guideline, rules (i.e., classification patterns) were manually summarized based on studying the positive instances in training data set.

In this study, a classification rule is considered as a description of relations between semantic units which are indispensable for identifying the presentation of a specific health care guideline. Considering the variation of natural language expression, we believe that a classifier can have multiple classification rules and each corresponds to a distinct expression pattern. Such patterns are extractable from positive sentences. The knowledge engineering for extracting patterns starts from identifying the theme-related semantic units, then try to find connections between them. For most patterns, usually more than one positive instance can be

```

<RulePattern>
<ruleID>6</ruleID>
<patAmount>3</patAmount>
<!-- "antidepressant", "side effect",
"vary", "not"(NEGPunit), proxim-
ity(2,3)=[EITHER,5] -> <Pattern>
<PID>1</PID>
<punitAmount>4</punitAmount>
<eID>1</eID>
<keyword>Antidepressive
Agents</keyword>
<tagType>MMTx-1</tagType>
<pos>N</pos>
<synset>
<synCount>3</synCount>
<syn>
<term>MAOIs?</term>
<tagType>TEXT</tagType>
<pos>N</pos>
</syn>
<syn>
<term>SSRIs?</term>
<tagType>TEXT</tagType>
<pos>N</pos>
</syn>
<syn>
<term>SNRIs?</term>
<tagType>TEXT</tagType>
<pos>N</pos> </syn>
</synset>
<term>SSRIs?</term>
<tagType>TEXT</tagType>
<pos>N</pos>
<term>SNRIs?</term>
<tagType>TEXT</tagType>
<pos>N</pos>
</alternative>
<term>Pharmaceutical
Preparations</term>
<tagType>Hypernym</tagType>
<pos>N</pos>
</alternative>
<!-- This MMTx includes free text Medi-Y
cation, medicine and drug ->
<alternative>
<term>drug</term>
<tagType>Hypernym</tagType>
<pos>N</pos>
</alternative>
</alter_in_context>
<enforce>1</enforce>
<co-occurrence>
<co-flag>N</co-flag>
</co-occurrence>
</punit>
<punit>
<eID>2</eID>
<keyword>effect side</keyword>
<tagType>LVG</tagType>
<pos>N</pos>
<synset>
<synCount>1</synCount>
</synset>
</co-occurrence>
</co-occurrence>
</punit>
<punit>
<eID>3</eID>
<keyword>vary</keyword>
<tagType>LVG</tagType>
<pos>V</pos>
</synset>
<synCount>2</synCount>
<syn>
<term>change</term>
<tagType>LVG</tagType>
<pos>V</pos>
</syn>
</synset>
<term>alter</term>
<tagType>LVG</tagType>
<pos>V</pos>
<alter_in_context>
<altCount>1</altCount>
</alternative>
<term>differ</term>
<tagType>LVG</tagType>
<pos>V</pos>
</alternative>
</alter_in_context>
<enforce>1</enforce>
<co-occurrence>
<co-flag>Y</co-flag>
<co-termContainer>
N
</co-termContainer>
</co-occurrence>
</punit>
<punit>
<eID>3</eID>
<keyword>vary</keyword>
<tagType>LVG</tagType>
<pos>V</pos>
</synset>
<synCount>2</synCount>
<syn>
<term>change</term>
<tagType>LVG</tagType>
<pos>V</pos>
</syn>
</synset>
<term>barely</term>
<tagType>LVG</tagType>
<pos>N</pos>
</alternative>
<term>rarely</term>
<tagType>LVG</tagType>
<pos>N</pos>
</alternative>
</alter_in_context>
<enforce>1</enforce>
<co-occurrence>
<co-flag>Y</co-flag>
<co-termContainer>
Y
</co-termContainer>
<co-term>
<co-eid>3</co-eid>
<co-occur_proximity>
4
</co-occur_proximity>
<position_relation>
BEFORE
</position_relation>
</co-term>
</co-occurrence>
</punit>
</Pattern>
...
</Pattern> </RulePattern>

```

Fig. 3. Example of classification rules for health care guideline #6

found from training data. Features that commonly exist in the positive instances are used to define the classification pattern. The features that were used in this study include semantic tag, co-occurrence, part-of-speech, distance between key tags, tag ordering relations, and negative proposition.

Figure 3 lists out the XML-formatted classification rules for guideline #6. It has 3 patterns that were identified from positive training instances. Each pattern is comprised of multiple semantic units. For example, the content listed between the pair of XML markup <Pattern> and </Pattern> describes the first classification pattern. Key semantic units in this pattern include antidepressant, side effect, vary and so on. These required semantic units are called patternUnit.

The identity of a patternUnit is defined using LVG or MMTx tag. Under each patternUnit, constraints in terms of distance between a pair of patternUnits and the sequence of their occurrences are defined to describe the co-occurrence relationship between patternUnits, whenever such relationship exists. In order to classify a semantic tag instance, which represents a sentence, into TRUE, the specified constraints need to be satisfied. The example here is that the semantic unit “vary” needs to co-occur with another unit, “side effect”, either BEFORE or AFTER, with interval terms or phrases no more than 5. These values are configured based on statistics from training instances.

The classification working logic is simple. It reads in the semantic tag instance of a sentence; then matches it against the pre-defined classification patterns for a specific health care guideline. During the matching, computer scan the semantic units within a semantic tag instance to search for patternUnits required by the classification and calculate the matching probability based on the searching result. If any fitting pattern is confirmed, then classification result is TRUE, otherwise FALSE.

Quality Scores: Given a web page, its quality rating score is automatically generated based on the sentence classification results. The classifiers classify the test instance (i.e., sentence) as either Positive or Negative regarding to specific guidelines. If it is positive, the webpage containing the sentence scores one. However, the presentation of a same guideline in one web page will be counted only once. This complies with the standard used human rating. So, the final quality score is equal to the number of unique guidelines that rule-based classifiers identified in a web page.

4 Quality Rating Test

Data: This study testifies the semantics-based automated quality rating approach using depression treatment web pages. The whole corpus includes 201 depression treatment web pages (see Appendix A of [21]) was collected in 2009 from three types of sources as listed in Table 1. For search engines, “depression treatment” was used as the query and the first 30 returned web pages from each search engine were collected as candidates. For web portals, candidate pages were collected from depression treatment related sections only. Candidate pages were examined manually to remove duplicate pages and pages that were inappropriate for other reasons (see Appendix B of [21]). In the end, 201 web pages were selected to form the corpus.

All 201 pages were rated by human raters with reference to a set of evidence-based depression treatment guidelines (see Appendix C in [21]) previously used in [7,8]. Human rated quality scores for all these pages range from 0 to 8. The pages were divided into five bins according to scores, i.e., 0, 1–2, 3–4, 5–6, and 7–8. Stratified random sampling was conducted to get 31 pages as testing data set, and the remaining 170 pages formed training data.

Measures for Classification Performance: The testing data set has 31 web pages and in total 2677 sentences. Precision, recall and accuracy were used to

Table 1. Sources for constructing the corpus

Web search engines	Medical search engines	Health care web portals
Google	AOL OmniMedicalSearch	Medline Plus in United States
Yahoo! Search	HealthFinder	HealthlinkBC in Canada
Microsoft Bing Search	HealthLine	HealthInsite in Australia
Ask.com	MedNar	National Health Service (NHS)
HealthFinder	WebMD	in United Kingdom

evaluate the sentence classification performance. The measures are calculated using the following equations: Precision = $TP / (TP + FP)$, Recall = $TP / (TP + FN)$, Accuracy = $(FP + TN) / (TP + FN + FP + TN)$. TP stands for the number of true positive (cases) identified by classifier; FP stands for false positives identified by classifier; FN stands for false negatives identified by classifier; and TN stands for true negatives identified by classifier.

Classification Results: Each guideline has a dedicated classifier, and the classification performance is shown in Table 2. Overall, the accuracy of rule-based classifiers is very high (>99.4%). Recall ranges from 75% to 100%. The variation of recall across guidelines may be attributed to a variety of factors including the number of ways to paraphrase a specific guideline, the number of available positive training cases, and the coverage of different paraphrasing patterns in the training data.

Therefore, we also used micro-averaging to combine the above values into one quantity in order to measure the classification performance across different guidelines. By micro-averaging, classifiers (for each guideline) are allowed to participate in performance evaluation equally. The micro-averaging results are listed in the last row in Table 2, with 78.3% precision, 82% recall, and 99.8% accuracy.

Quality Rating Results: The classification performance listed above attests to the effectiveness of semantics-based automated quality rating. Table 3 shows the quality scores assigned by both rule-based automated rating and human rating. Automated rating scores are pretty close to human rating scores. 45.2% of testing pages have the same score, 32.3% and 19.3% of pages have one score lower or higher than human rated score. Only one page (testing page #15) has rule-based score higher by 3.

The rule-based rating results are very close to human rating results in terms of not only the number of identified criteria in each web page, but also the accuracy of identification. Given the 31 testing pages, human raters identified 92 unique guidelines. Rule-based classifiers identified 91 unique guidelines. 78 identified items are the same between the two sets. 83.7% of the human identified depression treatment guidelines were also identified using the rule-based rating system. Only 16.1% of the human identified guidelines were missed; and 14.3% of computer identified guidelines were false positives.

The ultimate quality rating performance is measured using Pearson correlation between automatically rated scores and human rated scores. Given the 31

Table 2. Performance of sentence classification by rule-based classifiers

Rating criteria	Human classification	Rule-based classification (Y)	Rule-based classification (N)	Recall	Precision	Accuracy
#1	Y N	40 7	9 2621	81.6%	85.1%	99.4%
#2	Y N	3 4	0 2670	100.0%	42.9%	99.9%
#3	Y N	0 0	0 2677	NA	NA	100.0%
#4	Y N	0 0	0 2677	NA	NA	100.0%
#5	Y N	0 0	0 2677	NA	NA	100.0%
#6	Y N	16 3	3 2655	84.2%	84.2%	99.8%
#7	Y N	5 4	1 2667	83.3%	55.6%	99.8%
#8	Y N	2 0	0 2675	100.0%	100.0%	100.0%
#9	Y N	1 1	0 2675	100.0%	50.0%	100.0%**
#11	Y N	6 0	2 2669	75.0%	100.0%	99.9%
#12-A	Y N	9 1	2 2665	81.8%	90.0%	99.9%
#12-B	Y N	10 3	3 2661	76.9%	76.9%	99.8%
#13-A	Y N	4 0	0 2673	100.0%	100.0%	100.0%
#13-B	Y N	2 0	0 2675	100.0%	100.0%	100.0%
#14	Y N	14 4	4 2655	77.8%	77.8%	99.7%
#15	Y N	2 6	0 2669	100.0%	25.0%	99.8%
#20	Y N	9 1	3 2664	75.0%	90.0%	99.9%
Micro-averaging	Y N	123 34	27 45325	82.0%	78.3%	99.9%

Table 3. Quality score assigned to testing web pages by rule-based auto-rating system

Testing Page ID	Quality score via human rating	Quality score via rule-based rating	Quality score difference
1	7	7	0
2	7	6	-1
3	8	7	-1
4	6	5	-1
5	6	6	0
6	5	5	0
7	5	4	-1
8	4	5	1
9	3	4	1
10	4	3	-1
11	3	4	1
12	3	4	1
13	4	4	0
14	3	2	-1
15	2	5	3
16	2	3	1
17	2	2	0
18	2	2	0
19	2	2	0
20	3	2	-1
21	2	2	0
22	2	1	-1
23	2	1	-1
24	1	2	1
25	1	1	0
26	1	1	0
27	1	1	0
28	1	0	-1
29	0	0	0
30	0	0	0
31	0	0	0
Total	92	91	Not applicable

testing pages, the Pearson correlation is positive and significant, with r equal to 0.909. r^2 equals to 0.827. That means 82.7% of the variance of the rule-based quality scores is associated with the variance in the evidence-based human rated scores.

5 Discussion

Precision and recall indicate the ability of the automated approaches to correctly identify positive instances of each criterion. The higher recall, the fewer actual criteria sentences go undetected (lower false negative rate). The higher precision, the fewer non-criterion cases are mistakenly identified as a criterion (lower false positive rate). Results in Table 2 shows that the lowest recall was 75%, and the average recall was 82% across whole guideline set. This suggests that the semantics-based rule classification system can be fairly effective in identifying the presentation of depression treatment guidelines in web text.

Guideline #1:

“Antidepressant medication is an effective treatment for major depressive disorder.”

False Positive Match:

10. The test, called the cytochrome P450, helps pinpoint genetic factors that influence your response to certain antidepressants (as well as some other medications).

Fig. 4. A false positive example

On the other hand, Table 2 shows that precision of sentence classification varies in a wide range across different guidelines. This indicates that the classification rules defined for certain guidelines need to be enhanced with more distinguishing constraints for filtering out false positives. In addition, strongly skewed data was part of the reason for low precision. The negative over positive ratios for all criteria is averagely 302:1. Particularly, for guidelines (i.e., #2, 6 and 15) which have low precisions, the negative over positive ratio ranges from 445:1 to 1337:1. That means true positives (TP) can be easily overwhelmed by false positives (FP) even though only a very small percentage of actually negative cases are mistakenly identified as positive, hence precision can be low.

Pearson correlation is used to evaluate the quality rating scores. The strong positive correlation with human rating results suggests that the computer rated scores predicated the content quality of depression treatment web pages in a manner close to human being performance. The low precision and imperfect recall of sentence classification did not seem to have greatly affected the page rating performance. This is partly related to a fact that a single guideline is commonly paraphrased more than once in a web page. Among multiple presentation of a same guideline in a page, as long as one presentation is captured by automated classification system, the quality score is added by one without being hurt by false negatives. Similarly, suppose that the classifier label five sentences

as presentations of a guideline, with four being false positives, i.e., precision = 20%. Because there is one sentence being a true positive, the impact of false positives would not be reflected in the automatically assigned quality rating score.

Although 83.7% of human identified guidelines were captured by the automated rating system, there is space for improvement to make both false positive and false negative errors lower. It is found in the case review of classification results that false positive errors occur when the semantics of a text segment is taken for the entire sentence. For example (see Fig. 4), because the sentence contains “your response to certain antidepressant”, the classifier mistakenly classified the sentence as a match for guideline #1. To avoid false positives like this, the classification rules need to be supplemented with strict description logic.

Another limitation of the current implementation is that it uses individual sentences as processing unit while between-sentence analysis (e.g., co-reference) has not been utilized. For this reason, false negatives happened in a few situations in which the meaning of a guideline is expressed across multiple sentences, typically in bullet list format (see Fig. 5). After sentence splitting, “exercise” and “depression” were separated into two different sentences. Hence neither of them was predicated as a presentation of guideline #20.

Guideline #20:

“Exercise can be effective for depression.”

False negative (missing) sentence:

Regardless of whether you have mild or major depression, the following self-care steps can help:

Get enough sleep.

Follow a healthy, nutritious diet.

Exercise regularly.

Fig. 5. A false negative example

Guideline #6: “The side effect profile varies for different antidepressants.”

Testing sentences identified as positive cases:

1. Overall, no type of antidepressant drug is more effective than any other, but the different types can have different side effects, and different drugs sometimes are more or less effective for different individuals.
2. The side effects vary depending on the type of antidepressant you take.
3. SSRIs and SNRIs are more popular than the older classes of antidepressants, such as tricyclics—named for their chemical structure—and monoamine oxidase inhibitors (MAOIs) because they tend to have fewer side effects.
4. However, because TCAs tend to have more numerous and more severe side effects, they’re often not used until you’ve tried SSRIs first without an improvement in your depression.

Fig. 6. Examples of correctly classified sentences

In spite of the limitations, the performance testing results suggest the effectiveness of the semantics-based automated quality rating approach in two aspects. First, by semantics-based classification, this study converts quality rating task to a task of identifying health care guideline presentation among web text, following a procedure similar to human rating. However, our approach is automated since computer can rely on semantics-based classification rules to distinguish positive instances (i.e., guideline presentation) from negative ones. The classification is considered semantics-based since both classification input and classification rules are generated based on semantics. Among training data, various positive training cases are semantically categorized into different groups depending on their way for paraphrasing a same treatment guideline. Features commonly existing in a same group are extracted to form a classification rule, which just corresponds to an expression pattern. Therefore, a classifier with well-trained classification rules can identify the presentation of a guideline in different expression patterns. Figure 6 shows some identified positive sentences. These examples include different ways for expressing guideline #6. Pattern a) says that side effects of antidepressants are “different”; pattern b) uses “vary” to paraphrase; pattern c) indicates variation by a discussion of “fewer/more” side effects between antidepressants. In the listed cases, the rule-based classifier successfully identified that the sentences are in concordance with the rating guideline #6.

It has to be acknowledged that training data set may not necessarily cover all expression patterns used in human communication. Thus, it is possible that a developed has incomplete classification rule set and hence can miss some positive cases in testing. Statistically speaking, however, patterns with high frequency of usage would more likely be learned from training data set than those with low frequency. Thus, the impact of missing pattern on classification recall can be controlled reasonably low by preparing the randomly sampled training data set of reasonably large size.

Second, the shallow semantic analysis and the generated semantic representation of sentences turned to be generically effective for classification tasks relative to different treatment guidelines. Through transforming text content from English natural language to semantic tag instance in our defined syntax, sentence semantics are kept and conveyed in an appropriate sufficiency for supporting classification. It is also important that the semantic tagging process in this study was independent from the treatment guidelines in that no processing was customized to deal with any specific guidelines and its unique content and concepts.

6 Conclusion

This study proposed a semantics-based approach for implementing automated quality rating on web health care pages according to evidence-based health care guidelines. Web pages with depression treatment content are used for case study. The experimental results show that the automatically generated quality scores

have strong and positive correlation with human rated scores. That is, automatically generated quality scores have potential to be valid indicators of the quality of depression treatment web pages. Different from previous research, this automated approach is semantics-based, with aim to rely health care information quality rating directly on content. Through shallow semantic analysis and semantics-based classification, computer could identify the presentation of health care guidelines with reasonable accuracy (in Tables 2 and 3).

In the current implementation, the rule-based classifier utilized expression patterns manually extracted from training data to empower automated binary classification of sentences in untouched data set. Hence, human efforts for reading text and identifying the presentation of guidelines among enormous amount of web pages can be avoided. In the future we will explore the use of machine learning to enhance the automation of pattern learning process as well. In addition, we will also attempt to apply this semantics-based approach to rate the content quality of web pages in other health conditions. If the results of this study are replicable and generalizable, this automated quality rating approach could add significant value to the quality assessment practice of health care information on the web.

References

1. Barnes, C., Harvey, R., Wilde, A., et al.: Review of the quality of information on bipolar disorder on the internet. *Aust. N. Z. J. Psychiatry* **43**(10), 934–945 (2009)
2. Burkell, J.: Health care information seals of approval: what do they signify? *Inf. Commun. So.* **7**(4), 491–509 (2004)
3. Chen, L.E., Minkes, R.K., Langer, J.C.: Pediatric surgery on the internet: is the truth out there? *J. Pediatr. Surg.* **35**(8), 1179–1182 (2000)
4. Crocco, A.G., Villasis-Keever, M., Jadad, A.R.: Analysis of cases of harm associated with use of health care information on the internet. *J. Am. Med. Assoc.* **287**(21), 2869–2871 (2002)
5. Eysenbach, G., Powell, J., Kuss, O., Sa, E.: Empirical studies assessing the quality of health care information for consumers on the world wide web. *J. Am. Med. Assoc.* **287**(20), 2691–2700 (2002)
6. Frické, M., Fallis, D.: Verifiable health care information on the internet. *J. Educ. Libr. Inf. Sci.* **43**(4), 246–253 (2002)
7. Griffiths, K.M., Christensen, H.: Website quality indicators for consumers. *J. Med. Internet Res* **7**(5), e55 (2005). <http://www.jmir.org/2005/5/e55/>
8. Griffiths, K.M., Tang, T.T., Hawking, D., Christensen, H.: Automated assessment of the quality of depression websites. *J. Med. Internet Res.* **7**(5), e59 (2005). <http://www.jmir.org/2005/5/e59/>
9. HONcode: Honcode: Principles — quality and trustworthy health care information (2012). <http://www.hon.ch/HONcode/Conduct.html>
10. Khazaal, Y., Chatton, A., Zullino, D., Khan, R.: Hon label and discern as content quality indicators of health-related websites. *Psychiatr. Q.* **83**(1), 15–27 (2012)
11. Kiley, R.: Does the internet harm health? some evidence does exist that the internet harms health. *BMJ* **323**(7331), 328–329 (2002)

12. Kunst, H., Groot, D., Latthe, P., Latthe, M., Khan, K.S.: Accuracy of information on apparently credible websites: survey of five common health topics. *BMJ* **321**(7337), 581–582 (2002)
13. Martin-Facklam, M., Kostrzewa, M., Martin, P., Haefliger, W.E.: Quality of drug information on the world wide web and strategies to improve pages with poor information quality: an intervention study on pages about sildenafil. *Br. J. Clin. Pharmacol.* **57**(1), 80–85 (2003)
14. Pew Internet and American Life Project: Internet health resources (2003). <http://www.pewinternet.org/~media/Files/Reports/2003/PIP%20Health%20Report%20July%202003.pdf>
15. Pew Internet and American Life Project: Online health search 2006 (2006). <http://www.pewinternet.org/~media/Files/Reports/2006/PIP%20Online%20Health%202006.pdf>
16. Pew Internet and American Life Project: The social life of health care information (2011). <http://pewinternet.org/Reports/2011/Social-Life-of-Health-Info.aspx>
17. Podichetty, V.K., Booher, J., Whitfield, M., Biscup, R.S.: Assessment of internet use and effects among healthcare professionals: a cross sectional survey. *Postgrad. Med. J.* **82**, 274–279 (2006)
18. Silberg, W.M., Lundberg, G.D., Musaccio, R.A.: Assessing, controlling, and assuring the quality of medical information on the internet: caveat lector et viewer—let the reader and viewer beware. *J. Am. Med. Assoc.* **277**(15), 1244–1245 (1997)
19. Smith, A.: Evaluation of information sources the world-wide web virtual library. <http://www.vuw.ac.nz/staff/alastair%20smith/avaln/evaln.htm> (2002)
20. Wang, Y., Liu, Z.: Automatic detecting indicators for quality of health care information on the Web. *Int. J. Med. Inform.* **76**, 575–582 (2007)
21. Zhang, Y.: Semantics-based automated quality assessment of depression treatment web documents. Ph.D. thesis, The University of Western Ontario, London, Ontario, Canada (2012)



OntoSenseNet: A Verb-Centric Ontological Resource for Indian Languages

Jyoti Jha^(✉), Sreekavitha Parupalli, and Navjyoti Singh

Center for Exact Humanities, IIIT-Hyderabad, Hyderabad, India
{jyoti.jha,sreekavitha.parupalli}@research.iiit.ac.in,
navjyoti@iiit.ac.in

Abstract. Following approaches for understanding lexical meaning developed by Yāska, Patanjali and Bhartrihari from Indian linguistic traditions and extending approaches developed by Leibniz and Brentano in the modern times, a framework of formal ontology of language was developed. This framework proposes that meaning of words are *in-formed* by intrinsic and extrinsic ontological structures. The paper aims to capture such intrinsic and extrinsic meanings of words for two major Indian languages, namely, Hindi and Telugu. Parts-of-speech have been rendered into sense-types and sense-classes. Using them we have developed a gold-standard annotated lexical resource to support semantic understanding of a language. The resource has collection of Hindi and Telugu lexicons, which has been manually annotated by native speakers of the languages following our annotation guidelines. Further, the resource was utilised to derive adverbial sense-class distribution of verbs and kāraka-verb sense-type distribution. Different corpora (news, novels) were compared using verb sense-types distribution. Word Embedding was used as an aid for the enrichment of the resource. This is a work in progress that aims at lexical coverage of language extensively.

Keywords: Ontological structures · Hindi · Yasca

1 Introduction

The concept of ‘meaning’ has been discussed for a long time. Cognitively it can be understood to have an *intensional* or *extensional* form. Frege [1] discussed the idea of sense and reference. He called ‘sense’ as *intensional* meaning and ‘reference’ as *extensional* meaning. The meaning that has a constant value in an expression is *intensional*, whereas the meaning that is contributed by the real world to the mental concept is *extensional*. Two words are said to be extensionally equivalent if they refer to the same set of objects, whereas if they share the same features then they are intensionally equivalent. According to Frege

J. Jha—Contributed to resource development for Hindi

S. Parupalli—Contributed to resource development for Telugu.

© Springer Nature Switzerland AG 2023

A. Gelbukh (Ed.): CICLEing 2018, LNCS 13397, pp. 32–45, 2023.

https://doi.org/10.1007/978-3-031-23804-8_3

every significant linguistic expression has both 'sense' and 'reference'. The other theories of meaning are correspondence theory, consensus theory, constructivist theory etc. All these account for extensional meaning.

Meaning of a word in a language is generally derived from dictionary or from a context it is used in. Speaking from an ontological viewpoint, the meaning of a word can be understood based on its participation in classes, events and relations. In order to manipulate language computationally at the level of lexical meanings, Otra [2] developed Formal Ontology of Language. It considers meaning to have an intrinsic form. According to the theory proposed, meanings have primitive ontological forms. It is language independent and aims at extensive coverage of language.

This paper uses the idea of Formal Ontology of Language to develop lexical resource for Hindi and Telugu. Section 2 discusses the previous works that have been done in order to specify meaning of a word and development of various lexical resource. Section 3 of the paper discusses the Formal Ontology of Language, as proposed by Otra [2]. Section 4 talks about data acquisition for Hindi and Telugu and it shows how sense identification is done for different parts-of-speech. *kāraka* information for sense-types of verbs have been extracted from Hindi corpus. OntoSenseNet, a user interface has been built for our ontological resource. Section 5 shows validation of the resource based on inter-coder agreement. Section 6 demonstrates enrichment of the resource using word embeddings. Representation of verbs through their adverbial class distribution has been studied. Different corpora (news, novels) have been compared using frequency profiling of verb sense-types. Section 7 outlines conclusion and future work. IAST based transliteration¹ for Devanagari and Telugu scripts has been used in the paper.

2 Previous Work

In this section we discuss several attempts that have been made by various researchers in order to specify meaning of a word. Considering verb as the core of a language, some linguists derived different classifications. Along with different kinds of verb classifications, there have been approaches to derive semantic primitives.

Levin's Classification. Levin assumed that syntactic behavior of a verb is semantically determined [3]. He classified meanings of about 3,000 English verbs. They were composed into 50 primary-classes and 192 sub-classes using methodical study of 79 diathesis alternations. It mainly deals with verb taking noun and prepositional phrase complements. Although it can empirically classify verbs but it only captures some facets of semantics [4]. It does not include verbs taking ADJP, ADVP, predicative, control and sentential complements and is highly language dependent.

VerbNet Classification. VerbNet [5] is a hierarchical verb lexicon that represents verbs syntactic and semantic information. In each verb class, thematic

¹ https://en.wikipedia.org/wiki/International_Alphabet_of_Sanskrit_Transliteration.

roles are used to link syntactic alternations to semantic predicates. However, it contains limited coverage of lemmas and for each lemma the coverage of the senses are limited. Since it has been inspired from Levin’s verb classification it is also language dependent.

WordNet. It is a lexical database inspired by psycholinguistic theory of human lexical memory [6]. Words are organized as synsets. These synsets represent lexicalised concepts which are organized into synonym sets. These synsets are connected to other synsets by means of semantic relations. Nouns are organized as hypernymy and hyponymy relations. Verbs are organized as hypernym, troponym, entailment and coordinate terms. It does not have classification of adverbs. It also lacks information about verb syntax and is also language specific.

Wierzbicka’s Semantic Primitives. The concepts that can be innately understood without any further decomposition are Semantic Primes. The widely used example to explicate this concept is the verb ‘*touching*’. Its meaning can be readily understood, however a dictionary might define ‘*touch*’ as “to make contact” and ‘*contact*’ as “touching”, provides no information if neither of these words are understood. The theory of semantic primes was introduced by Wierzbicka, [7]. It has been criticised for its reductive approach and is limited by its generative coverage in any language [8].

The limitations of the above theories in terms of being language specific and limited coverage in a language led to the formulation of Formal Ontology of Language by Otra [2]. In the proposed theory, the meaning are considered to have primitive ontological forms and they are independent of a language. Each of the parts-of-speech are organised as *types* or *classes*. To derive the intensional meaning of a sentence, one has to consider the relation between different parts-of-speech e.g. the relation between verb-adverb, noun-adjective, verb-noun. The relations between the points are also considered to have ontological forms, which can help specifying meaning at a sentence level. The next section discusses the theory of Formal Ontology of Language, as introduced by Otra.

3 Formal Ontology of Language

In a language one can describe a state of affairs using different verbs, hence there is a verbal ambiguity. To derive the universal verb there have been several discussions in Greek and Indic traditions. While in Greek tradition ‘be’ is considered as the universal verb [9], on the other hand Indic tradition considers ‘happening (bhavati)’ as the universal verb. Let us consider a question “what are you doing?”. This can be answered with the verb ‘do’. Hence one can say that ‘do’ can be a primitive sense that will be present in every verb. However, Patanjali mentions three verbs that cannot be the answer to the above question. These are (1) being/existence (asti), (2) presence (vidyate), (3) happening (bhaāva). According to Bhartrihari, verb has sense of sequence and state. Hence, it has a sense of happening making it the universal verb. Linguistic traditions in India have long regarded verb as the centre of language (in both syntactic and semantic terms)

right from Yāska, Pānini, Patanjali and Bhartrihari [10–12]. Meaning of verbal element is seen as *bhāva* (happening) as opposed to *sattā* (being) which stands behind nominal elements [13, 14]. Later, independent of linguistic discourse, logicians brought out hundreds of *bhāva-s* (happening) with atomic transformational structure $\langle \text{entity}_1 | \text{entity}_2 \rangle$ like cause|effect, part|whole, predecessor|successor, qualifier|qualified, ascribed|ascriber, locus|located, etc. These are atomic discriminants which form elementary meanings. Meanings are not seen as an entity like semantic primitive [7] but as a unified discriminative structure with a form $\langle \text{entity}_1 | \text{contiguous with} | \text{entity}_2 \rangle$, in context of *continuum*. For example, verb ‘move’ has a sense $\langle \text{predecessor state} | \text{contiguous with} | \text{successor state} \rangle$, in context of ‘move’ *continuum*. Its meaning is a continuant feel of motion punctuated by discriminating logical structure of predecessor and successor states. One can read in its meaning such discrimination points. Leibniz called such punctuations as actual points [15] as endeavours, as ontologically vacuous, as different from Euclidean points. Brentano [16] also built an idea of mental continuants as punctuated with *modo recto* and *modo obliquo*. These boundaries, punctuations or points are ontological as they vacuously discriminate ontic entities or states which are felt as continuous. Otra [2] built formal ontology of lexical meaning using such punctuational boundaries.

Meaning of ‘move’ is always more than the discriminative senses we read in it. In the proposed formal ontology seven discriminant punctuations that are read in all verbs [2] are suggested and determined. When we say, ‘rapidly move’ or ‘hesitantly move’, we have done adverbial modification of the meaning of ‘move’ and have added new modifier|modified points in its meaning. When appending ‘rapidly’ we add temporal-feature-class in adverb whereas while appending ‘hesitantly’ we add force-feature-class in adverb to the meaning of ‘move’. Even when we have discriminated temporal or force features, meanings of ‘rapidly’ and ‘hesitantly’ are more than their adverb sense-classes. Otra [2] has delineated four adverb classes of discriminant point. Verbs are also seen as contiguous of nouns in seven or eight case relations. These seven/eight classes of verb-noun pairing are further coincident boundaries in the meaning of articulation with the verb. Further, noun-noun pairs and noun-adjective pairs are more coincident points. Otra [2] also proposes twelve noun-verb types of sense-points in the ontology. The verb-centric formal ontology of meaning is based on sense-type and sense-class of punctuational boundaries that can be located in lexical meaning. TYPES and CLASSES are logical forms of intensional senses [[2], page 13, 14, 15]. Using the formal ontology we are building lexical resource of verbs, adverbs, nouns and adjectives in terms of basic discriminant points, which *in-form* their meaning. The formal ontology is language independent and thus we are developing the resource for several languages at once.

In the next section we discuss the resource creation for Indian languages, namely Hindi and Telugu using the proposed Formal Ontology of Language.

4 Resource Building

Otra [2] has developed the resource for English that has 3,867 verbs, 1,980 adverbs and 300 adjectives. In our resource, Sense-types of 3,152 Hindi and 3,379 Telugu verbs has been manually identified. Similarly manual identification of sense-classes of 2,214 Hindi and 101 Telugu adverbs has been done. Sense-types of 238 Hindi adjectives has been identified. Annotation for sense-types of Telugu adjectives is in progress. The annotators have native proficiencies in the corresponding languages.

4.1 Data Acquisition

Words were collected from different resources like dictionary, wordnet. These were further used to populate our resource. Since this is a work in progress, not all the words from the different resources have been added into our resource.

Hindi. Distinct verbs, adverbs and adjectives for Hindi were collected from Hindi Wordnet² and Dictionary³.

Telugu. Telugu being a resource poor language, does not have a usable soft copy of Telugu-Telugu dictionary till date. We are developing it from the printed copy of “Sri Suryaraayandhra Telugu Nighantuvu” [17] for all of its eight volumes. Verbs, adverbs and adjectives have been completely populated in the usable soft copy from this dictionary, whereas work is still under progress for other parts-of-speech. Dictionaries for Telugu-Hindi, English-Telugu are available⁴.

Table 1 shows the number of distinct verbs, adverbs and adjectives in each of the resources.

Table 1. Distinct number of verbs, adverbs and adjectives for Hindi, Telugu in different resources

Resource	Distinct verbs	Distinct adverbs	Distinct adjectives
Hindi Wordnet	6778	2114	19190
Hindi-Shadsagara Dictionary	3529	1650	36398
OntoSenseNet (Hindi)	3152	2214	238
Telugu-Telugu Dictionary	8483	253	11305
Telugu Wordnet ^a	2795	442	5776
Telugu-Hindi	9939	142	1253
OntoSenseNet (Telugu)	3379	101	<i>Annotations are yet to be started</i>

^a<http://tdil-dc.in/indowordnet/>

² <http://www.cfilt.iitb.ac.in/wordnet/webhwn/>.

³ <https://ia601603.us.archive.org/20/items/in.ernet.dli.2015.348711/2015.348711.Hindi-Shabdasagar.pdf>.

⁴ https://ltrc.iiit.ac.in/onlineServices/Dictionaries/Dict_Frame.html.

4.2 Sense-Identification

Verbs. Different verbs can be used for describing the same situation. Thus verbs are collocative in nature. In a single verb many verbal sense points can be present and different verbs may share same verbal sense points. For example “walking”, “running” entails a sense of ‘move’. Verb like “studying” entails a sense of ‘know’ and ‘do’. “Eating” entails a sense of ‘do’ and ‘have’. Thus, verbs are organised as sense-types. Otra [2] has shown the existence of seven primitive sense-types of verbs. These seven sense-types of verbs have been derived by collecting the fundamental verbs used to define other verbs. These verbs were then grouped using intrinsic senses, which were designated to a particular sense-type. These sense-types are inspired from different schools of Indian philosophies. The seven sense-types of verbs are listed below with their primitive sense along with two Hindi and Telugu examples each.

1. Means|End - Do; khelanā (play), karanā (do); āduta (play), ceyuta (do)
2. Before|After - Move; bahanā (flow), calanā (walk); pāruta (flow), naduvuta (walk)
3. Know|Known - Know; jānanā (know), parakhanā (examine); ūhimcuta (imagine), parisīlimcuta (examine)
4. Locus|Located - Is; rahanā (stay), honā (happen); umduta (to be, stay), jaruguta (happen)
5. Part|Whole - Cut; kātanā (cut), mitānā (erase); koyuta (cut), vidipovuta (separate)
6. Wrap|Wrapped - Cover; jhāmpānā (cover), pahanānā (dress-up someone); mūyuta (cover), ākramimcuta (contain forcefully)
7. Grip|Grasp - Have; pānā (get), lenā (take); bhayapaduta (fear), tīsukonu (take)

Each of the verbs can have all the seven dimensions of sense-types. The degree depends on the usage/popularity of a sense in a language. In our resource we have identified two sense-types of each verb, i.e. primary and secondary. Consider the verb ‘dance’ in the sentence “Madhuri is dancing gracefully”. Here ‘dance’ involves a sense of movement which a doer does. Thus Before|After is a primary sense and Means|End is a secondary sense. For polysemous verbs, sense-type identification was done for each of their different meanings. For example, the verb “rap” has three meanings. Thus rap1, rap2, rap3 have been added in the resource along with its meaning sense-types.

1. rap1- Criticizing someone, Means|End and Know|Known.
2. rap2- To perform rap music, Means|End and Before|After.
3. rap3- To hit or say something suddenly and forcefully, Means|End and Part|Whole.

Sense-types for 3,152 Hindi and 3,379 Telugu verbs were manually identified. Figure 1 shows the sense-type distribution for English, Hindi and Telugu verbs in OntoSenseNet.

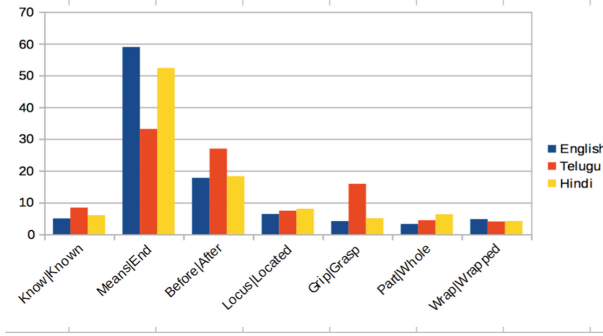


Fig. 1. Verb Sense-type distribution

Adverbs. Meaning of verbs can further be understood by adverbs, as they modify verbs. The sense-classes of adverbs are inspired from adverb classification in Sanskrit. Following are the identified sense-classes along with their fundamental sense, illustrated with English, Hindi and Telugu examples

1. Temporal - Adverbs that attributes to sense of time. e.g. Never; sasamaya (timely); varusagā (continuously)
2. Spatial - Adverbs that attributes to physical space. e.g. There; pās (near); davvu (far away)
3. Force - Adverbs that attributes to cause of happening e.g. Dearly; barbas (unwillingly); gattiga (tightly)
4. Measure - Adverbs dealing with comparison, judgement. e.g. - Only;; lagbhag (approximately); gaddu (abundantly)

Sense-classes for 2,214 Hindi and 101 Telugu adverbs have been manually identified. Table 2 shows sense-class distribution of adverbs for English, Hindi and Telugu.

Table 2. Adverb sense-class distribution

Sense-class	English	Hindi	Telugu
Temporal	5.5	24.3	28.7
Spatial	2.7	13.5	12.8
Measure	39.4	32.2	31.6
Force	52.2	30	26.7

Sub-classification of adverb sense-classes is being developed.

Kāraaka Relation. Kāraaka are classes that are used for expressing relation between words in a sentence. Computational Paninian Grammar Framework describes kāraakas as syntactico-semantic (or semantico-syntactic) relations between the verbs and their related constituents (generally nouns) in

a sentence [18]. It describes eight types of kārakas:- k1: kartā (Nominative), k2: karma (Instrument), k3: karna (Ablative), k4:sampradāna (Possessive), k5:apādān (Objective), k6:sambandh (Dative), k7:adhikaran (Locative), k8:sambodhan (Vocative). Distribution of verb sense-types and kāraka were studied in Hindi novel corpora containing 3,39,057 words. This corpus was collected from Hindisamay⁵ and was fully parsed using ISCNLP tagger⁶. For Telugu, development of treebank data and full dependency parser is still under process. Thus, Kāraka-Verb sense-types distribution of Telugu has not been extracted. Figure 2, shows verb sense-types distribution for different kāraka in Hindi. It shows that Locus|Located type of verbs have mostly occurred with a k1 relation with noun, whereas the k4 kāraka is hardly occurs in any verb-noun relation.

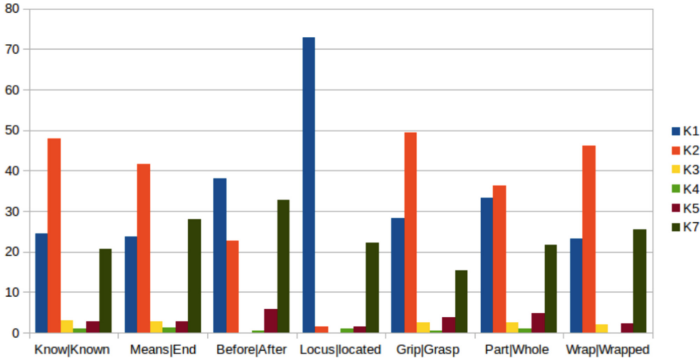


Fig. 2. Verb sense-type and Karaka distribution

Adjectives. Like verbs, adjectives are also collocative in nature. Otra [2] identifies 12 sense-types. However these can be reduced to 6 pairs. Following are the identified six sense-types pairs of adjectives along with their meanings and one English, Hindi and Telugu examples each.

- Locational - Adjectives that universalise or localise a noun e.g. Local, doosrā (Other), nirdista (specific)
- Quantity - Adjectives that either qualify cardinal measure or quantify in ordinal-type e.g. - only, eka (one), okkati (One)
- Relational - Adjectives that qualify nouns in terms of dependence or dispersal e.g. similar, mātrihīn (without mother), vistrta (broad)
- Stress - Adjectives that intensify or emphasis a noun - e.g, strong, mazbūt (strong), gatti (strong)

⁵ <http://www.hindisamay.com/>.

⁶ <https://github.com/iscnlp/iscnlp>.

- Judgement - Adjectives that qualify evaluation or qualify valuation feature of a noun e.g. - bad, acchā (good), mamci (good)
- Property - Adjectives that attribute a nature or qualitative domain of a noun. e.g. - black, kālā (black), nallani (black)

Sense-types for 238 Hindi adjectives have been manually annotated. Sense-types for Telugu is yet to be started.

4.3 User Interface

A user interface, OntoSenseNet has been developed for this ontological resource. Input of a verb/adverb/adjective returns its corresponding sense-types/classes along with its illustrative meaning and example sentences. Further, development on crowdsourcing of the resource is being done where users can login and populate the data by providing examples to support their claims. This will be manually verified before adding to OntoSenseNet. For divided opinion, a discussion page would be provided.

5 Resource Validation

To show the reliability of the resource, Cohen’s Kappa [19] was used to measure inter coder agreement. The annotation was done by one human expert and it was cross-checked by another annotator who was equally trained. Verbs and adverbs were randomly selected from our resource for the evaluation sample. The inter coder agreement for 500 Hindi verbs and 1,000 adverbs were 0.70 and 0.91 respectively. Similarly validation for 500 Telugu verbs was done, for which inter coder agreement was 0.82. Validation for both the language resources shows high agreement [20]. Further validation of the resource is in progress.

6 Resource Enrichment and Utilization

6.1 Sense-Identification of Verbs and Adverbs Using Word Embeddings

Word Embeddings have been widely used for extracting similar words [21]. Previous study has shown that word embedding has significant improvement over WordNet based measures [22]. We used this property to assign sense-type of verbs and sense-class of adverbs. This was done in order to facilitate the annotation task. However, this was further verified manually.

Method. Hindi corpus was collected from Leipzig⁷, Hindi wiki-dump⁸ and Lindat [23]. It contains 3,73,45,049 sentences and 75,31,64,082 words. This corpus was fully parsed using iscnlp tagger⁹. Word2vec [24] was trained on this corpus

⁷ <http://corpora2.informatik.uni-leipzig.de/download.html>.

⁸ <http://kperisetla.blogspot.in/2013/01/wikipedia-offline-wikipedia-in-hindi.html>.

⁹ <https://github.com/iscnlp/iscnlp>.

using CBOW technique and vector dimensions were 100.1,485 verbs and 1,054 adverbs were randomly chosen from the corpus for the sense-identification. Out of these, sense-type of 1,182 verbs and sense-class of 832 adverbs were already present in the resource. The sense identification for the remaining words were carried out. In order to identify sense of a word, its cosine similarity was calculated against the words whose sense were already present in the resource. Cosine similarity above 0.7 was considered. The maximum occurring sense in the similarity cluster was considered to be the potential sense of that word. This was subsequently verified manually. For example, sense-type of the Hindi verb 'cīranā' (tear) was not present in the resource (OntoSenseNet). The sense-type of those verbs were considered whose cosine similarity with 'cīranā' was above 0.7. The maximum occurring sense from these set of verbs was Part|Whole. Thus, the sense-type of 'cīranā' was assigned as Part|Whole. The above method was executed to identify sense-type of 303 verbs and sense-class of 222 adverbs. This method correctly identified the sense-class of 220 adverbs and sense-type of 185 verbs. The sense identified for the words were finally incorporated in the resource. Table 3 summarises the statistics. Column A is part-of-speech. Column B shows number of words in that part-of-speech that were randomly sampled from the corpus. Column C shows number of words for which sense were already present in the resource. Column D shows number of words for which sense identification was carried out. Column E contains number of words whose sense were correctly identified by Word2Vec. Column F shows accuracy in percentage.

Table 3. Statistics for the sense-identification by Word2Vec

A	B	C	D	E	F
Verb	1,485	1,182	303	185	61.056%
Adverb	1,054	832	222	220	99.09%

Table 4 and Table 5 shows the verb and adverb clusters, respectively. In each of the tables the similarity with the words in column-1 is above 0.7. Column 3 shows the maximum occurring sense-type/sense-class.

Table 4. Similarity cluster and the maximum occurring sense-type

Verb	Verb-clusters	Maximum occurring sense-type
cīranā	nocanā, ghisanā, chedanā, khuracanā, pīsanā, phulānā	Part Whole
jānanā	batānā, kahanā, lenā-denā, mālūma, mānanā, pūchanā	Know Known

Table 5. Similarity cluster and the maximum occurring sense-class

Adverb	Adverb-clusters	Maximum occurring sense-class
tigunā	dogunā,dugunā,caugunā	Measure
yakāyaka	sahasā,ekāēka,acānaka	Temporal

6.2 Representation of Verbs Through Adverbial Semantics

Representation of verbs as a combination of their participatory adverb modifiers has not been exhaustively studied till now. Using our resource one can study the adverbial features of verb. We extracted Verb-Adverb relation from a fully parsed corpora.

Using full dependency parser of Hindi, 25,00,130 sentences were parsed. Verbs whose frequency was above 50 were considered in order to extract their modifying adverbs. The sense-class of these adverbs were then identified using our resource. Percentage of the frequency distribution of these sense-classes of adverbs for every verb was calculated. Table 6 shows few examples of representation of verbs in terms of their frequency distribution of adverb sense-class.

Table 6. Percentage of frequency distribution of adverb sense-class of verbs

Verb	Temporal	Measure	Spatial	Force
cunanā	60.82	36.08	2.06	1.03
jānā	61.12	25.92	12.96	0
calanā	44.26	44.26	6.55	4.91
likhanā	32.07	50.31	10.69	6.91

Few examples of the verbs that were not modified by “Spatial” in the corpora are karwāne [to make someone do], chaunk [to be surprised], bachā [save], gher [circle] It is interesting to note that spatial and force were the only classes that did not modify some verbs. OntoSenseNet has verb-adverb pairing frequencies also.

Corpora Comparison. Frequency distribution of verb sense-type and adverb sense-class across different types of corpora(novel, news) have been statistically studied. Previous works have shown the usage of frequency profiling [25] for comparing different corporas. We have used this approach to identify key ontological points that differ across corpora. Log-Likelihood estimation was calculated using contingency table for each of the verb sense-type. It was observed that Means|End sense-type is the most indicative of the news corpora that was collected using Hindi Treebank(3,65,431 words).

Table 7 shows the frequency distribution of sense-types of verbs and their log-likelihood.

Table 7. Frequency distribution and Log Likelihood

Sense-type	Novel	News	Log-Likelihood
Means End	25.215	38.270	+38523.04
Before After	19.084	15.293	+9787.00
Part Whole	5.917	5.736	+4290.64
Grip Grasp	7.387	9.782	+9076.68
Locus Locate	30.817	23.946	+14911.13
Know Known	10.216	5.993	+2812.73
Wrap Wrapped	1.360	0.977	+566.23

Furthermore, adverbial class distribution was observed in novels of two different authors. The adverbial distribution was considered for the most commonly occurring verbs in both the corpora. The difference in the use of adverbs may be accounted for different sociolinguistics aspects and can be applied in the study of social differentiation in the use of a language. Adjectival and kāraka information needs to be exploited further for a deeper insight into corpora comparison. Sense-identification for Telugu using Word2vec is in progress. Table 8 shows few examples of the adverbial sense-class distribution for the verbs used by both the authors.

Table 8. Adverbial sense-class distribution across different novels

Verb	Adverb sense-class for Author-1	Adverb sense-class for Author-2
letnā (To take rest)	Temporal	Temporal, Measure
khelnā (To play)	Temporal	Temporal, Measure, Force
likhnā (To write)	Measure	Temporal
girnā (To fall)	Temporal, Spatial, Force, Measure	Temporal, Force
jānā (To go)	Temporal	Temporal, Measure
denā (To give)	Temporal, Measure	Temporal
sunanā (To listen)	Measure	Temporal, Measure

7 Conclusion and Future Work

In this paper we used Formal Ontology of Language to develop ontological resource for Hindi and Telugu. Logical forms of intensional senses were identified as type and class. The validation of this resource was done using Cohen’s Kappa that showed higher agreement. The resource was used for extracting adverbial class distribution of verbs, kāraka-verb sense-type distribution from corpus. We compared different corpora based on sense-type distribution of verbs. Novels of different authors were compared using sense-class of adverbs. The usage of

different sense-class of adverbs across different authors indicates different sociolinguistics aspect. However, this just covers a portion of a language. Adjectival and kāraka points will give a deeper insight. Further, validation and enrichment of the resource is in progress. Major work ahead is to find etymological, morphological and syntactic points. The resource can be utilized for word sense disambiguation, synonymity measure, cultural studies.

References

1. Gamut, L.: *Logic, Language, and Meaning*, volume 1: Introduction to Logic. University of Chicago Press, Chicago (1991)
2. Otra, S.: *Towards building a lexical ontology resource based on intrinsic senses of words*. PhD thesis, International Institute of Information Technology Hyderabad (2015)
3. Levin, B.: *English Verb Classes and Alternations*, vol. 1. University of Chicago Press, Chicago (1993)
4. Korhonen, A., Briscoe, T.: Extended lexical-semantic classification of English verbs. In: *Proceedings of the HLT-NAACL Workshop on Computational Lexical Semantics*, pp. 38–45. Association for Computational Linguistics (2004)
5. Kipper, K., Dang, H.T., Palmer, M., et al.: Class-based construction of a verb lexicon. In: *AAAI/IAAI* **691**, 696 (2000)
6. Miller, G.A., Beckwith, R., Fellbaum, C., Gross, D., Miller, K.J.: Introduction to wordnet: An on-line lexical database. *Int. J. Lexicogr.* **3**, 235–244 (1990)
7. Wierzbicka, A.: *Semantic Primitives*, 2nd edn. pp. 134–137. Elsevier (1972)
8. Riemer, N.: Reductive paraphrase and meaning: a critique of wierzbickian semantics. *Linguist. Philos.* **29**, 347 (2006)
9. Kahn, C.H.: *The Verb “Be” in Ancient Greek*. Hackett Publishing Company, Indianapolis (1973)
10. Mimāṅsak, Y.: *Sanskrit Vyakaran Shastra ka Itihas*. Yudhishtir Mimāṅsak (1985)
11. Staal, J.F., Staal, F.: *A Reader on the Sanskrit Grammarians*. MIT Press, Cambridge (1972)
12. Potter, K.H.: *The Encyclopedia of Indian Philosophies, Volume 3: Advaita Vedanta Up to Samkara and His Pupils*. Princeton University Press, Princeton (2014)
13. Bhattacharya, B.: *Yāska’s Nirukta and the Science of Etymology: An Historical and Critical Survey*. Firma KL Mukhopadhyay, Calcutta (1958)
14. Ogawa, H.: *Process & Language: A Study of the mahabha, sya ad a1. 3.1 bhuvadayo dhatavah*. Motilal Banarsidass, Delhi (2005)
15. Leibniz, G.W.: *The Labyrinth of the Continuum: Writings on the Continuum Problem, 1672–1686*. Yale University Press (2001)
16. Brentano, F.: *Philosophical Investigations on Time, Space and the Continuum (Routledge Revivals)*. Routledge, London (2009)
17. Pantulu, J.: *Sri Suryarayandhra nighantuvu*. Andhra Pradesh Sahitya Akademi (1936)
18. Bharati, A., Sangal, R.: *Computational Paninian Grammar Framework, Supertagging: Using Complex Lexical Descriptions in Natural Language Processing*. p. 355. The MIT Press (2007)
19. Carletta, J.: Assessing agreement on classification tasks: the kappa statistic. *Comput. Linguist.* **22**, 249–254 (1996)

20. Landis, J.R.: Koch, G.G.: The measurement of observer agreement for categorical data. *Biometrics* **33**(1), 159–174 (1977)
21. Leeuwenberg, A., Vela, M., Dehdari, J., van Genabith, J.: A minimally supervised approach for synonym extraction with word embeddings. *Prague Bull. Math. Linguist.* **105**, 111–142 (2016)
22. Singh, S.B.R.P.D., Bhattacharyya, P.: Merging verb senses of Hindi wordnet using word embeddings. In: 11th International Conference on Natural Language Processing. p. 344 (2014)
23. : LAC Hindi corpus. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics, Charles University (2014)
24. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: *Advances in Neural Information Processing Systems*, pp. 3111–3119 (2013)
25. Rayson, P., Garside, R.: Comparing corpora using frequency profiling. In: *Proceedings of the Workshop on Comparing Corpora*, Association for Computational Linguistics, pp. 1–6 (2000)



Detection of Change in the Senses of AI in Popular Discourse

Ahmet Suerdem, Tugba Dalyan, and Savaş Yıldırım^(✉)

Istanbul Bilgi University, Santral Istanbul, Istanbul, Eyup, Turkey
{ahmet.suerdem, tdalyan, savasy}@bilgi.edu.tr
<http://www.bilgi.edu.tr>

Abstract. As chatbots, driverless cars and other robot-like applications become a part of everyday life, we are witnessing an increase in the popularization of Artificial Intelligence (AI) by the mass media. While this has some potential in terms of informing the public about technological development, it also makes the term a buzzword not pointing to any actual object with no agreed-upon meaning. AI is usually deployed as an umbrella term for sealing a variety of analytical tools such as intelligent decision support systems, deep learning, and computational linguistics disregarding their actual denotations. As the popular discourse and media represent its mundane features to connote miracles or apocalypses, AI gains a mythical status that can have different significations according to different cultural contexts. Our aim in this paper is to study the semantic shifts in the meaning of AI in different contexts by examining the mapping of the words to different semantic vector spaces over time.

Keywords: Word embeddings · Semantic shifts · Artificial intelligence

1 Introduction

Recently, the detection of semantic shifts in the meaning of words has gained considerable attention in the fields of information retrieval and computational linguistics. Semantic similarity can be measured according to a distributional model postulating that terms sharing similar contexts are semantically similar. First-order representations represent a word in a one-hot long vector in a word-by-word matrix in terms of co-occurrence statistics measuring the semantic similarity. On the other hand, recent advances in the literature suggest that second-order representations may overtake the former [1–3].

In that respect, Neural Network Language Models (NNLM) have demonstrated promising performance by reducing time complexity, especially for word representation learning. The most important characteristic of NNLM is their capacity to generate dense and short word embeddings that are highly effective for finding semantic and syntactic regularities [4, 5].

In this paper, our primary goal is to detect the change in the meaning of “Artificial Intelligence” (AI) over time by using word embeddings. The corpus is collected from media articles taken from major UK newspapers by scraping the Lexis/Nexis¹ queries for the keyword “artificial intelligence”. Once learned word representation, semantic similarity could be easily measured by simple metrics in a static model. Detection of semantic shift requires dynamic analysis allowing us to track and detect the changes in the meaning of AI across time. Hence we measured semantic shifts in the sense of AI across the last five years through word embedding vectors. Applying the word2vec model to build word vectors, we tracked the changes according to both global and local word embeddings models. Finally, to show the shifts in the semantic of AI, we present our results as time-series patterns.

1.1 Related Works

In recent years, there have been a variety of computational studies on the language changes over time [6–9]. [6] proposed a distributional similarity approach to a relative-frequency-based method using the Google Books Ngram data from the 1960s and 1990s. In [7], they proposed three methods based on frequency to extract sudden change in word usage, syntactic times series over part of speech tag distribution and distributional times series over embedding space by using three different datasets, The Google Books Ngram Corpus, Amazon Movie Reviews and Twitter data. [8] proposed a method to monitor of vocabulary shifts over time proceeds as follows: using distributional semantic models to infer semantic spaces over time from time-stamped textual documents, constructing semantic networks by applying graph-based measures to calculate saliency of terms, and shifting the vocabularies over time.

[9] showed that using a linear transformation is effective to find semantic shifts over time and how distributional methods can reveal the two statistical laws (law of conformity and law of innovation) of semantic change. [10] monitored semantic fluctuations over more than 400 years using time-stamped word representations per decade. They also proposed a visual analytics framework for visualizing lexical change at three different levels - individual words, word pairs, and sentiment orientation. A similar approach is proposed to compute the semantic shifts using word embeddings trained on corpora that represent specific viewpoints and evaluated on political speeches and media reports [11].

2 Word Representation

Distributional approaches represent words in vector space models (VSM) for the NLP problems. For example, [12] represented the sense of a word as a real-valued vector by using co-occurrence statistics to measure the semantic similarity. It is based on the idea that if two words share similar neighboring words, they are

¹ <https://www.lexisnexis.com/hottopics/scholastic/>.

likely to be similar. The similarity between the vectors of the words is simply computed by cosine similarity and other metrics. The main disadvantage of this method is the size and sparsity of the matrix that is equal to the size of the vocabulary. As the dimension of the vector exceedingly increases, so does the computational complexity of the designed system.

The widely applied solution is the feature elimination in the preparation step. It discards non-informative terms based on some metrics using corpus statistics. The study [1] pointed that the term frequency could be informative. Some feature selection criteria such as chi-square (χ^2) are found very effective to find informative terms from corpus, [1–3]. Another technique is to reduce the dimensionality such as in Latent Semantic Indexing [13] (or Latent Semantic Analysis). This technique is applied to produce informative and short latent dimensions. It uses Singular Value Decomposition (SVD) as a method for building significant dimensions derived from a document-term matrix. It is a member of a method family that can approximate an N-dimensional matrix using fewer dimensions such as Principle Components Analysis (PCA), Factor Analysis, etc. [14–16].

Besides these dimension reduction techniques, NNLM has recently become widely used and demonstrated promising performance by reducing time complexity. The most important characteristic of NNLM is its capacity of generating dense and short embeddings, namely word embeddings [4, 5]. In the architecture of the neural network, each word is initially associated with a random vector. As a two-layer neural network processes textual corpus, the vectors are iteratively updated by applying stochastic gradient descent (SGD) where the gradient is measured by back-propagation. The objective is to predict the middle word using the surrounding words or vice versa. Thus, the prediction task is typically similar to multi-class classification where the soft-max function is used to compute class probability estimation. The network finally learns the embeddings for all words that appeared in the corpus by convergence.

As one of the most popular word embeddings models, word2vec model [4] showed how word embeddings were efficiently trained within two different architectures, namely Continuous Bag of Words (CBoW) and the Skip-Gram (SG). The architecture achieved both minimizing computational time complexity and maximizing model accuracy. The second model, GloVe, [5] proposed another word embedding model. It is based on matrix factorization and a new global log-bilinear regression model. These two popular word embedding models also proved that embeddings are very good at capturing syntactic and semantic regularities, using the vector offsets between word pairs. Another important approach is fastText that exploits subword information to construct word embeddings. Therefore, it is capable of handling out-of-vocabulary words [17].

3 Methodology

We propose a model that measures the change in the sense of AI through time. For building time series we have produced semantic similarities between AI and other words over time.

Our procedure is as follows:

Algorithm

```

Building Time Series(C, years):
  let C be a global corpus
  pre-processing(C)
  applyNPChunker(C)
  let C_year be a local corpus for each year
  globalModel= WordEmbeddings(C)
  V= buildVoc(C, globalModel)

  # Build Local Word Embedding for each year
  for each year in(2013,2017):
    localModel[year]= WordEmbeddings(C_year)

  # semantic similarities of terms w with AI over time
  TimeSeries=[]
  for each year in (2013, 2019):
    model=localModel[year]
    for w in V:
      TimeSeries.append((w,year), model.SemSim("AI",w))

  # Clustering Time Series
  cluster=HierCluster(Normalize(TimeSeries))

```

In order to measure the semantic shift of a word, representative time slot corpora are needed [7]. We have decided to start from 2013, a date when the term AI has started to creep into the popular press because of Siri, Google now and Cortana and especially their application to smartphones using natural language to answer questions, make recommendations, and perform actions. We have collected the corpus by means of automatically scraping Lexis/Nexis queries for retrieving the news articles in major UK newspapers containing the keyword “artificial intelligence” between 2013 and 2017. We balanced the corpus by randomly selecting an equal number of articles.

After collecting the corpus, some pre-processing steps such as: cleaning noisy terms, tokenization, sentence boundary detection, stop words removal have been applied. For entity detection, we segmented and annotated multi-word sequences by means of noun-phrase chunking. Collocation sets can be created according to some metrics depending on corpus overall statistics and n-gram statistics which in turn be used as a chunker where we used bigrams and tri-grams. With the n-gram chunker, we captured the noun phrases such as Artificial Intelligence, Big Data and lemmatized them. Besides using the entire (i.e. global) corpus to measure such statistics, we have also used the local (i.e. annual subset) corpora

to compute local statistics. First, n-gram chunkers were trained through global corpus statistics, and then they were applied to each local corpus.

After preprocessing the corpus, we have trained the word embeddings model using both global corpus and the other five local corpora. For each subset corpus, a separate word embeddings model was built to measure the semantic and other differences between the terms across time. To train the word embedding model, we used the word2vec model. The preprocessed and annotated textual data were consumed by the word2vec model with mostly default configuration where dimension size is 300, minimum word frequent threshold is 5 and the context window is 10. We divided our article corpus into five temporal subsets S year. We create a vocabulary V by selecting those terms that appear in each S year, local corpus, and are similar to the term AI. The terms whose global corpus frequency is less than 50 are eliminated. Finally, we constructed a time series data frame for word semantics and usage. We applied the time series clustering technique to dynamic data for tracking the change where the terms are grouped in terms of their characteristics such as losing similarities or gaining similarities with AI. We present our findings as plots visualizing the time series trends in similarities.

4 Experiments

4.1 Preliminary Analysis

The pre-processed corpus is summarized in Fig. 1. The terms are sorted according to their frequencies and plotted in the histogram. A quick examination of the ten most frequent words shows that all these terms (i.e. machine learning, big data, and virtual reality) denote “artificial intelligence” as a generic concept. These two figures verify Zipf’s law indicating that the frequency of any word should be inversely proportional to its rank in the table of word frequency. Thus, the most frequent word in corpus occurs roughly twice as often as the second most frequent word, and so forth. The corpus does not show any idiosyncrasies.

4.2 Word Usage Analysis

After examining the corpus, we evaluated the change in word usage by creating a year by word count matrix where each cell represents how many times a word appears in the corresponding year. To simplify the word space and represent it on a two-dimensional space, we used Correspondence Analysis. Correspondence Analysis (CA) is a multivariate dimensionality reduction technique designed to explore relationships among categorical variables and jointly represent the patterns in their categories [19]. This technique is especially effective for cross-tabulated data and is widely used across many disciplines such as social sciences, history, and psychology because of its ease of understanding for the non-technical audience. The following Fig. 2 shows the Correspondence Plot where columns (years) and the rows (words) are jointly mapped into a 2D space. The positioning

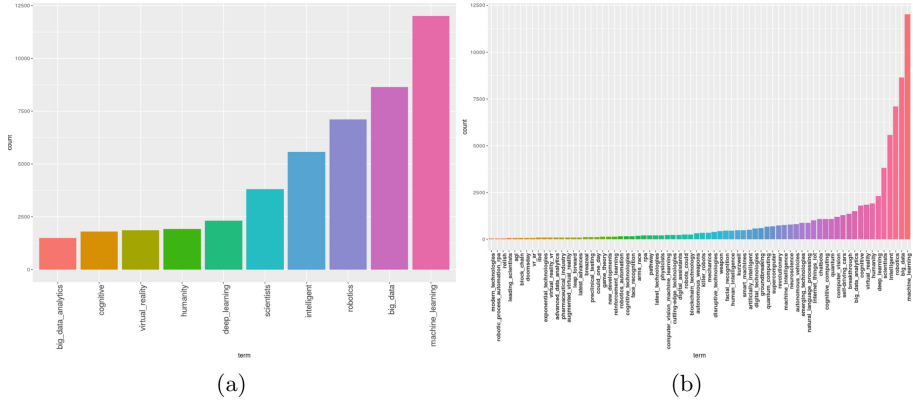


Fig. 1. A set of two subfigures describes: (a) Histogram of first ten terms; (b) Histogram of all terms

of words and years along the coordinates of this space represents Euclidean distances and nicely summarizes the groupings of years and words in terms of their semantic proximities. The circles in the figure show the marginal frequency of the word concerned. Greater circles represent more frequent words and since we have balanced the distribution of news articles by selecting the equal number of articles per year the sizes of the circle are comparable.

When we examine the plot, the years 2013 and 2014 are fairly close to each other, and words signifying more generic AI such as “big data” and “computers” are grouped around them. This suggests that the sense of AI is not yet much differentiated from the generic computer science and the term is perceived as a sub-discipline of information sciences. Yet, other years significantly diverge to signify a different agenda. For example, words like “machine intelligence”, “humanity”, “cognitive computing” group together around 2015 and some words from 2014 such as “doomsday”, “smart machines”, “human intelligence”, “weapons” and “doomsday” are close to this group. This suggests that like every new technology AI invokes the public imagination for utopian and dystopian fantasies. When we have a closer look into the articles in this group, we can see that they widely discuss the idea that humans will no more be the dominant species on earth and will be replaced by intelligent machines. Reference to doomsday scenarios like in the movie Terminator where the artificial intelligence Skynet becomes self-aware and starts a nuclear strike on humanity is prevalent in these articles.

Some of the articles are more optimistic, suggesting a more symbiotic relationship between intelligent machines and humans, and perceive AI as a great collaborator to reduce the labor burden on humanity. All in all, these articles commonly discuss more philosophical and existential issues about the effects of AI on the future of humanity rather than concrete applications. 2016 and 2017 largely diverge from the previous years and the words around them are mostly grouped at the right-hand quadrant of the map. The words like “disruptive

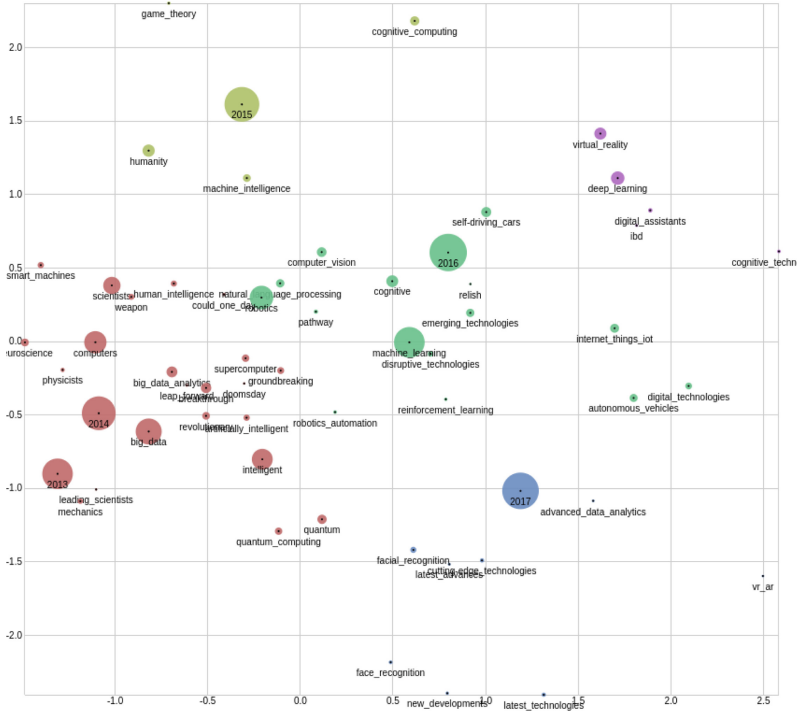


Fig. 2. Correspondence analysis of word usage

technologies”, “driverless cars”, “emerging technologies”, “machine learning”, “facial recognition”, “advanced data analytics” all signify prospects about the practical applications of AI and the changes they could bring to everyday life. Although 2016 and 2017 are quite distant on the map, this distance is because of the reference to the type of the technology rather than a substantive significance difference. While the hot topic for 2016 is “driverless cars”, “IOT” and “face recognition” underlines the year 2017. This suggests that each year new technology is launched to excite the public imagination about its prospective applications. When we make a deeper reading of representative articles, the articles largely discuss groundbreaking products displacing an established technology and creating a completely new industry.

4.3 Change in Meaning of AI Across Time

To monitor the semantic shifts in the sense of AI across time, we have first extracted a list of approximately eighty terms having the highest semantic similarity of word embeddings to AI from the entire corpus. This was needed to identify a lexicon representing the language of AI and facilitate the interpretation. Then, we have used these as a trimmed feature vector and recalculated the similarities of these terms to AI in each particular annual subset corpora. Hence,

we were able to detect which set of words converged or diverged to represent the sense of AI for that particular year. We have further reduced the word space into two dimensions by applying Principal Components Analysis (PCA) and mapped the distance of the term AI in each year to the words in the lexicon. Figure 3 shows that the second dimension shows more variability around opposite poles and hence more easy to interpret. The south-end of this dimension is populated by words such as “killer robots”, “humanity” and “Kurzweil” which represent more philosophical-speculative side of AI discourse. The north end, on the other hand, is populated by words such as “IOT”, “machine learning”, “big data analytics” which represent more concrete applications of AI. Hence, over the years, the sense of AI changes from more speculative to more concrete.

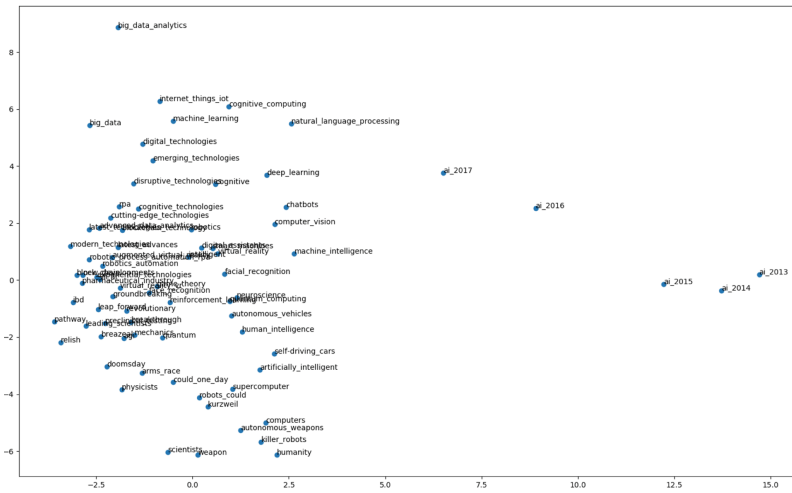


Fig. 3. PCA mapping of word embeddings for AI across years

Another interesting finding is that, while AI is quite distant from the overall lexicon in earlier years, it converges to the lexicon in the later years after 2015. While its sense stays more or less similar for the years before 2015 and distant to the global lexicon indicating a relative lexical poorness, we witness semantic shifts both in 2016 and 2017 consecutively towards the overall lexicon. We can interpret this as while AI did not have a steady language before 2016s, it has started to establish its own language with a specialized lexicon afterward. To monitor the evolution of the semantic distance of each term to AI, we grouped the time series of words into five meaningful clusters by means of hierarchical cluster analysis. The clusters group terms in terms of their similarities in changes across time (for time-series clustering techniques see [18]). This was a necessary step for a tidier representation of the evolution of the more than eighty words as can be seen from the messy Fig. 4. This helped us to focus on a deeper examination of the trend in semantic shifts.

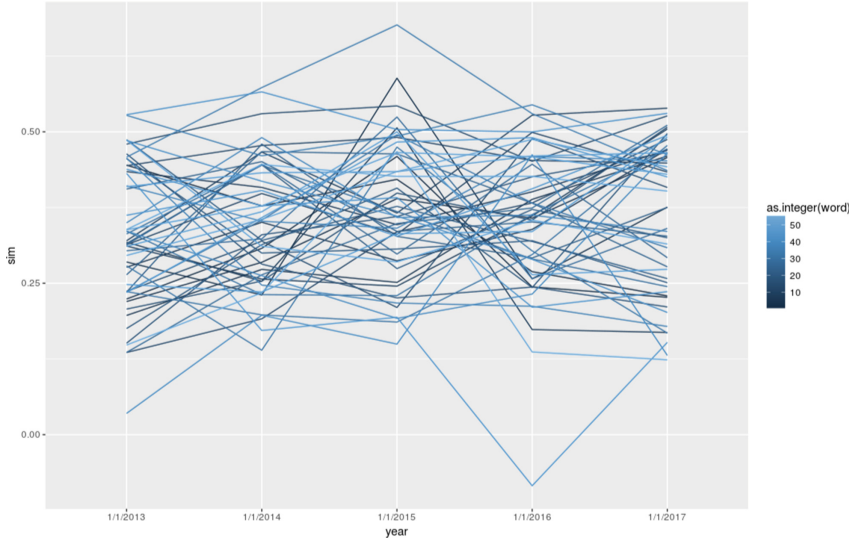


Fig. 4. Semantic similarities between all terms and AI over time

No matter how similar the words are to AI on average, those words showing the same patterns across years are clustered by normalizing the matrix rows and applying the Euclidian distance. The first cluster contains the words denoting particular AI techniques such as “big data”, “cognitive computing”, “natural language processing”, “reinforcement learning” as shown in Fig. 5(a). They show an increasing trend in terms of converging to AI and their final similarity scores are about 0.4 and over in 2017. The second cluster as shown in Fig. 5(b), depicts a declining trend in an opposite manner to the first one. This cluster contains generic science and computer terms such as “game theory”, “mechanics”, “quantum” and “super-computers”. This trend provides another evidence to our hypothesis that AI is distinguishing its language from that of generic science and establishing its own vocabulary. The third cluster is particular only to 2015 and is about the AI declaration by the experts we have mentioned earlier as we can see from the words it contains such as “arms race”, “autonomous weapons” and so on as shown in Fig. 5(c).

The fourth cluster in Fig. 5(d), shows a declining cyclic pattern and is also about the philosophical aspects of AI as we can observe from the words (Kurzweil, Humanity, human intelligence, neuro-science, ground-breaking) it contains. It represents more optimistic prospects about what AI would offer to humanity in the future. A close reading of the articles in this cluster presents interesting clues about prophetic claims about AI. Ray Kurzweil, a prominent futurist makes interesting claims about “transhumanism” which represents an intellectual movement aiming to transform the human condition by means of sophisticated technologies to greatly enhance human intellect and physiology. This sense of AI fluctuates and comes to the fore according to interesting events

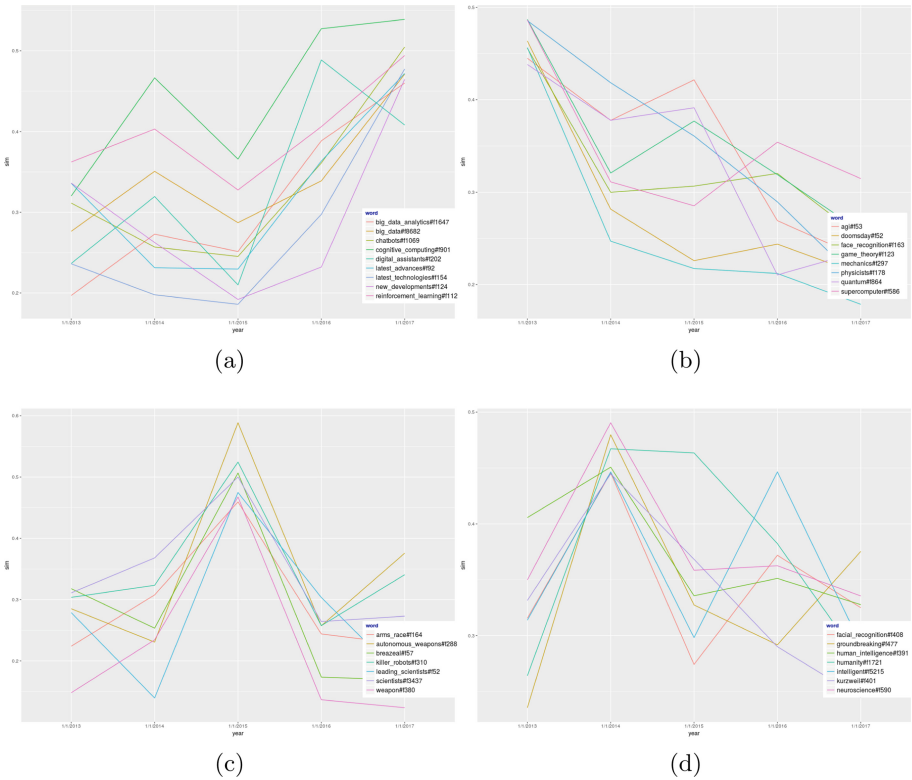
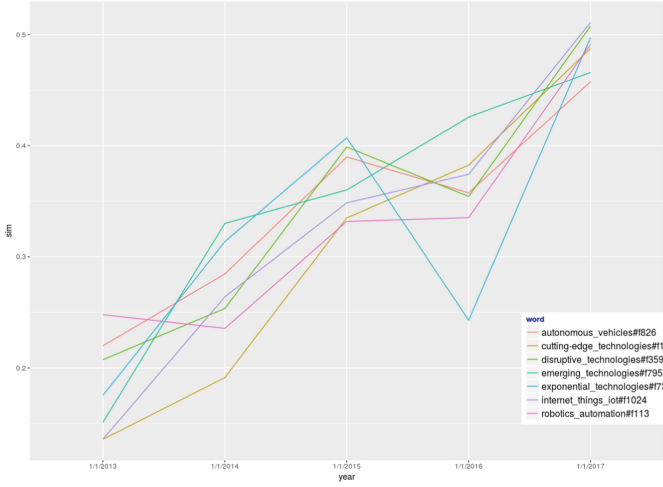


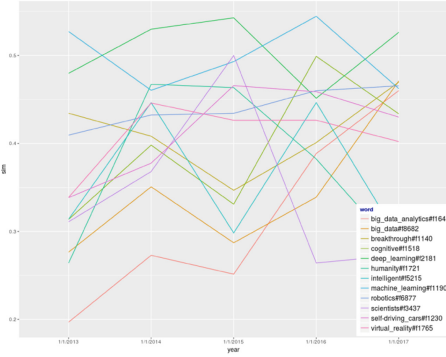
Fig. 5. A set of four subfigures describes: (a) Semantic similarities for first cluster; (b) Semantic similarities for second cluster; (c) Semantic similarities for third cluster; and, (d) Semantic similarities for fourth cluster

or declarations. The final cluster, in Fig. 6(a), shows a steadily increasing trend and represents the innovations and concrete applications of AI technology to everyday life (cutting-edge technologies, autonomous cars, IoT, robot automation, etc.) providing another evidence to the hypothesis that AI is evolving from a more speculative sense towards a more down to earth sense and is establishing its own language.

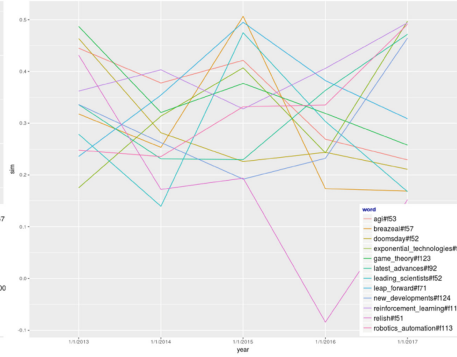
Finally, we checked the trend in most frequent and least frequent words. Time series is stationary and these terms do not show any trend or patterns in terms of their similarities to AI. This is understandable as these are either generic terms that might occur in every document or specific terms occurring in a few documents as plotted in Fig. 6(b, c).



(a)



(b)



(c)

Fig. 6. A set of four subfigures: (a) Semantic similarities for fifth cluster; (b) Semantic shift of most frequent words; and, (c) Semantic shift of low frequent words

5 Conclusion

Our aim in this paper is to study the semantic shifts in the meaning of AI in popular discourse by examining the mapping of the words to different semantic vector spaces over time. The corpus is collected from media articles taken from major UK newspapers. We have applied a variety of techniques to understand the change in the meaning of AI. While Correspondence Analysis is applied to plot the word usage across time, the word embedding model has been utilized to represent the semantic shift of the words. To monitor the changes in word embeddings, PCA is applied to reduce the dimensionality of embeddings and map the words in 2D space. We also use word embedding series to see the

change of the meaning of AI overtime where time series clustering is an important technique to understand the series of terms. This led us to interpret the change and to monitor the semantic shifts in the sense of AI across time. All these experiments showed that, over the years, the sense of AI changes from more speculative to more concrete. They also provided some pieces of evidence to our hypothesis that AI is distinguishing its language from that of generic science and establishing its own vocabulary and AI is evolving from a more speculative sense towards a more down-to-earth sense. In future work, we plan to mostly exploit transformer-based architecture to monitor language change since that transformer architectures have successfully dominated the field and since that they extract contextual word embeddings rather than a static one.

References

1. Schütze, H., Hull, D.A., Pedersen, J.O.: A comparison of classifiers and document representations for the routing problem. In: Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1995), pp. 229–237 (1995)
2. Lewis, D.D., Ringuette, M.: A comparison of two learning algorithms for text categorization. In: Symposium on Document Analysis and Information Retrieval, pp. 81–83 (1994)
3. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press, London (2008)
4. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. [arXiv:1301.3781](https://arxiv.org/abs/1301.3781) (2013)
5. Pennington, J., Socher, R., Manning, C.D.: Glove: global vectors for word representation. In: EMNLP, pp. 1532–1543 (2014)
6. Gulordava, K., Baroni, M.: A distributional similarity approach to the detection of semantic change in the Google Books Ngram corpus. In: Proceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics (GEMS 2011). pp. 67–71 (2011)
7. Kulkarni, V., Al-Rfou¹, R., Perozzi, B., Skiena, S.: Statistically Significant Detection of Linguistic Change. CoRR abs/1411.3315 (2014)
8. Kenter, T., Wevers, M., Huijnen, P., de Rijke, M.: Ad Hoc monitoring of vocabulary shifts over time. In: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, pp. 1191–1200 (2015)
9. Hamilton, W.L., Leskovec, J., Jurafsky, D.: Diachronic word embeddings reveal statistical laws of semantic change. In: ACL2016, pp. 1489–1501 (2016)
10. Jatowt, A., Duh, K.: A framework for analyzing semantic change of words across time. In: JCDL, pp. 229–238. IEEE Computer Society (2014)
11. Azarbondy, H., Dehghani, M., Beelen, K., Arkut, A., Marx, M., Kamps, J.: Words are malleable: computing semantic shifts in political and media discourse. In: CIKM 2017, pp. 1509–1518 (2017)
12. Salton, G.: The SMART retrieval system-experiments in automatic document processing. Prentice-Hall Inc., Upper Saddle River, NJ, USA (1971)
13. Deerwester, S., Dumais, S., Furnas, G., Landauer, T., Harshman, R.: Indexing by latent semantic analysis. *J. Am. Soc. Inf. Sci.* **41**, pp. 391–407 (1990)

14. Jurafsky, D., James H.M.: *Speech and Language Processing: An Introduction to Natural Language Processing, Speech Recognition, and Computational Linguistics*. 2nd edn. Prentice-Hall (2009)
15. Jolliffe, I.T.: *Principal Component Analysis*, Springer, Berlin (2002). https://doi.org/10.1007/978-3-642-27497-8_5
16. Stephen, R.B., Jonathan, M.C.: The role of factor analysis in the development and evaluation of personality scales. *J. Personal.* **54**, 106–148 (2007)
17. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. *Trans. Assoc. Comput. Linguist.* **1**(5), 135–46 (2017)
18. Aghabozorgi, S., Shirkhorshidi, A.S.: Time-series clustering - a decade review. *Inf. Syst.* **53**, 16–38 (2015)
19. Greenacre, M., Blasius, J.: *Correspondence Analysis and its Interpretation*. In: *Correspondence Analysis in the Social Sciences*. Academic Press, London (1994)



Sparse Word Representation for RNN Language Models on Cellphones

Chong Ruan^{1,2}(✉) and Yanshuang Liu²

¹ School of Electrical Engineering and Computer Science, Peking University, Beijing 100871, China

pkurc@pku.edu.cn

² Kika Tech, Longshaoheng Mansion, Hepingli South Road, Dongcheng District, Beijing 100013, China

yanshuang.liu@kikatech.com

Abstract. Language models are a key component of input methods, because they provide good suggestions for the next candidate input word given previous context. Recurrent neural network (RNN) language models are the state-of-the-art language models, but they are notorious for their large size and computation cost. A main source of parameters and computation of RNN language models is embedding matrices. In this paper, we propose a sparse representation-based method to compress embedding matrices and reduce both the size and computation of the models. We conduct experiments on the PTB dataset and also test its performance on cellphones to illustrate its effectiveness.

Keywords: Language model · Recurrent neural network · Word embedding · Sparse representation · Input method

1 Introduction

Language modelling is a fundamental task in natural language processing, and can also be combined with other tasks such as spelling correction, machine translation and speech recognition. RNN language models [1, 2] are capable of utilizing arbitrarily long history in theory, making them an ideal choice for language modelling. Despite RNN's powerful modelling capacity, its large size limits its application: the size of such models will easily grow to tens of megabytes or even larger, which is cumbersome for mobile or embedded devices.

Researchers have proposed many techniques to compress large neural networks, including weight pruning [3, 4] and weight sharing [5, 6]. Nevertheless, weight pruning leads to irregular connection patterns in the final pruned model, making it unfriendly to hardware; and weight sharing techniques often involve modifying standard neural network structures and sometimes impose extra constraints on training algorithms, making them difficult to incorporate with standard RNN layers. In [7] it has been observed that a large portion of parameters in RNN language models comes from the embedding matrix and proposed to

compress the embedding matrix using sparse representation. Thus this method does not affect the network architecture and can be easily combined with popular RNN cells such as LSTM [8, 9].

In this paper, we follow the same sparse representation approach as in [7] but go even further: we aim to reduce both the size and the computation cost of RNN language models. **Our main contributions are three-folds:**

- We propose a binary search procedure to ensure each vector is represented by a fixed number of basis vector, thus better exploits parallel processing capabilities of the hardware;
- We derive a re-formulation of logits computation which can be combined with sparse representation perfectly and reduces computation cost;
- We test our model’s performance and availability on cellphones instead of just performing theoretical analysis.

The rest of this paper is organized as follows: In Sect. 2, we review the background of our work. Our proposed method is illustrated in Sect. 3 and experiment results are presented in Sect. 4. Finally, Sect. 5 concludes our work and discusses further directions.

2 Related Works

2.1 Skip-gram Word Vector

Word embeddings, also known as word vectors, are dense and low dimensional representations of words. One of the most popular tool to train word embeddings is *word2vec* [10]. We briefly summarize skip-gram model with negative sampling as follows:

$$L = \log \prod_{w \in C} \prod_{c \in \text{Context}(w)} g(w, c) = \sum_{w \in C} \sum_{c \in \text{Context}(w)} \log g(w, c) \quad (1)$$

$$g(w, c) = \log \sigma(E_w^T O_c) + \sum_{c_N \in \text{NEG}(w)} [\log \sigma(-E_w^T O_{c_N})] \quad (2)$$

where (w, c) is a word-context pair, $\text{NEG}(w)$ is the set of negative samples for word w , $\sigma(t) = 1/(1 + \exp(-t))$ is the sigmoid function, $E, O \in \mathbb{R}^{n \times |V|}$ are input and output embedding matrices, and $E_x, O_y \in \mathbb{R}^n$ are input embedding for word x and output embedding for word y respectively. In many literature, only matrix E is regarded as word embeddings. However, as argued by [11], we use both matrix E and O in our experiment, which leads to a more effective way of initializing RNN language models.

2.2 RNN Language Model

RNN language models read a word embedding E_{w_t} as their input at each time step t , do some internal computation $f(\cdot)$, and predict the distribution of the next word \mathbf{o}_t :

$$\mathbf{h}_t = f(E_{w_t}, \mathbf{h}_{t-1}) \quad (3)$$

$$\mathbf{o}_t = \text{softmax}(P\mathbf{h}_t + \mathbf{c}), \quad (4)$$

where $E \in \mathbb{R}^{n \times |V|}$ and $P \in \mathbb{R}^{|V| \times n}$ are input and output embedding matrix respectively, $\mathbf{c} \in \mathbb{R}^n$ is a bias term. In our experiment, we will use LSTM as the default RNN cell $f(\cdot)$.

2.3 Sparse Representation

The sparse representation idea is to exploit the redundancy in embedding matrices: embedding matrix can be viewed as $|V|$ vectors in \mathbb{R}^n . Considering $|V| \gg n$, one can choose a group of over-complete bases in \mathbb{R}^n and approximate each word vector using a sparse linear combination of basis vectors, which leads to a compression method. Since the number of parameters in embedding matrices is much larger than that in hidden layers ($O(n|V|)$ vs. $O(n^2)$), compressing embedding layers compresses the entire model.

The formulation in [7] is as follows: Denote basis matrix by $U \in \mathbb{R}^{|B| \times n}$ (each column of U is a basis vector in \mathbb{R}^n , and all $|B| > n$ columns form a group of over-complete bases), a word vector by \mathbf{w} , they determine basis weights $\mathbf{x} = (x_1, x_2, \dots, x_{|B|})$ as the solution of the following optimization problem:

$$\min_{\mathbf{x}} \|U\mathbf{x} - \mathbf{w}\|_2^2 + \alpha \|\mathbf{x}\|_1 + \beta |\mathbf{1}^T \mathbf{x} - 1| + \gamma \mathbf{1}^T \max\{\mathbf{0}, -\mathbf{x}\}, \quad (5)$$

where the first term is the approximation error, the second term controls the sparseness of weights \mathbf{x} , the third term requires the sum of all weights to be close to 1, and the last term favors non-negative weights. While these regularization terms have their intuition, they introduce 3 additional hyper-parameters and make it more difficult to optimize.

They simply choose the word vectors of the most frequent words as the over-complete basis vectors and solve the equation above for all word vectors of infrequent words to obtain the sparse codebook. Because many components in \mathbf{x} are zeros, one just need to store the indices and values of non-zero weights.

3 Methodology

This section consists of two subsections. In the first subsection, we describe the algorithm for learning sparse codings, the key part of which is a binary search procedure to ensure each sparse coding is of fixed length; in the second subsection, we illustrate how to utilize this sparse representation to reduce computation during prediction.

3.1 Proposed Sparse Representation

Our sparse representation technique is similar to the one in [7], but we simplify it to a basic LASSO problem and make it more tractable. We also assume words are sorted by their frequency in descending order, so that for a embedding matrix $E \in \mathbb{R}^{n \times |V|}$, its first $|B|$ columns $U = E_{1:|B|}$ are word vectors of the most frequent $|B|$ words, where $|B| > n$ is a hyper-parameter specified manually.

The Proposed Algorithm Tries to Represent a New Word Vector w by a Linear Combination of *exactly* s Basis Vectors. It has four inputs: an over-complete basis matrix $U \in \mathbb{R}^{n \times |B|}$, a new word vector w to be approximated, an integer s which indicates the desired sparseness, and a float tolerance tol used in terminating condition. And it returns two values: *indices*, an integer array of length s , denoting the ids of chosen basis vectors; and *weights*, a float array of length s , containing coefficients of the linear combinations.

The pseudo code of our algorithm is demonstrated in Algorithm 1.1. LASSO is used to control sparseness, but because we don't know the optimal regularization strength α^* which can give us an exactly s -hot solution x^* , we set up a large range of the optimal α^* , and reduce this range by iterated trials. This binary search procedure converges very quickly.

When the range is small enough, we probably have obtained a good enough α_t , so we break it in line 13 and gather the indices and values of non-zero entries in current x^* . Note that there is a possibility that the number of non-zero entries in x^* is slightly fewer than pre-specified s (because we break the loop in the strong regularization branch), we need to add more basis vectors with zero weights to embedding w 's sparse representation to force a fixed length approximation, which is the "Zero padding" part of the algorithm.

For each column vector $E_j \in \mathbb{R}^n$ in the whole embedding matrix E , we can pass E_j as parameter w in Algorithm 1.1 and learn a sparse representation for it. Run over all $|V|$ columns in E , we can get $|V|$ indices and weights and stack all them up into two matrices of shape $s \times |V|$, as illustrated in Fig. 1.

Compression Ratio: The compression ratio is $n \times |V| / (n \times |B| + 2s \times |V|)$. Suppose $n = 400, s = 10, |B| = 2000$ and $|V| = 20000$, which is indeed a practical setting we used in our mobile experiment, the ratio is 6.67. Noting that the elements in index matrix I are all non-negative integers less than $|B|$, we can use even fewer bits to store this matrix. Output embedding matrix can be compressed similarly with each matrix in Fig. 1 transposed.

It is also very easy to restore a word vector from the basis vectors and its sparse representation: one just need to fetch proper basis vectors and add them up with corresponding weights, as in Algorithm 1.2.

3.2 Fast Prediction

To predict the next word, an RNN language model need to read current word embedding. For standard RNN language model, the current word embedding

Algorithm 1.1. Sparse Code Learning Algorithm

```

1: procedure LEARN-SPARSE-CODING( $U, \mathbf{w}, s, tol$ )
    ▷ Choose  $s$  vectors from columns of  $U$  to approximate word embedding  $\mathbf{w}$ 
2:    $\alpha_{min} \leftarrow 1e-3$ 
3:    $\alpha_{max} \leftarrow 1e3$ 
                                        ▷ Binary search
4:   while true do
5:      $\alpha_t = (\alpha_{min} + \alpha_{max})/2$ 
6:      $\mathbf{x}^* \leftarrow \min_{\mathbf{x}} \frac{1}{2n} \|U\mathbf{x} - \mathbf{w}\|_2^2 + \alpha_t |\mathbf{x}|_1$ 
7:      $k \leftarrow \text{NUMBER-OF-NON-ZERO-ENTRIES}(\mathbf{x}^*)$ 
8:     if  $k > s$  then
9:        $\alpha_{min} \leftarrow \alpha_t$ 
10:    else
11:       $\alpha_{max} \leftarrow \alpha_t$ 
12:      if  $\alpha_{max} - \alpha_{min} < tol$  then
13:        break
14:      end if
15:    end if
16:  end while
                                        ▷ Extract non-zero entries from  $\mathbf{x}^*$ 
17:  indices  $\leftarrow \text{INDICES-OF-NON-ZERO-ENTRIES}(\mathbf{x}^*)$ 
18:  weights  $\leftarrow \text{VALUES-OF-NON-ZERO-ENTRIES}(\mathbf{x}^*)$ 
                                        ▷ Zero padding
19:  if  $k < s$  then
20:    Randomly choose  $s - k$  column ids from basis in  $U$ 
21:    Append these  $s - k$  ids to indices
22:    Append  $s - k$  zeros to weights
23:  end if
                                        ▷ Now both indices and weights have exactly  $s$  elements.
24:  return indices, weights
25: end procedure

```

Algorithm 1.2. Word Embedding Restoring Algorithm

```

1: procedure RESTORE-WORD-EMBEDDING( $U, \text{indices}, \text{weights}$ )
    ▷ Restore a word vector from its of sparse representation form
2:    $\mathbf{v} \leftarrow 0$ 
3:   for  $i = 1..\text{len}(\text{indices})$  do
4:      $\mathbf{v} \leftarrow \mathbf{v} + \text{weights}[i]U_{\text{indices}[i]}$ 
5:   end for
6:   return  $\mathbf{v}$ 
7: end procedure

```

can be fetched directly from the input embedding matrix. For our sparse representation, one need to use Algorithm 1.2 to recover the embedding and feed it to RNN, which involves s embedding lookups, vector scaling and vector addition and thus increases model computation slightly. However, **we can use the shared bases to speedup the computation of the output side, a main**

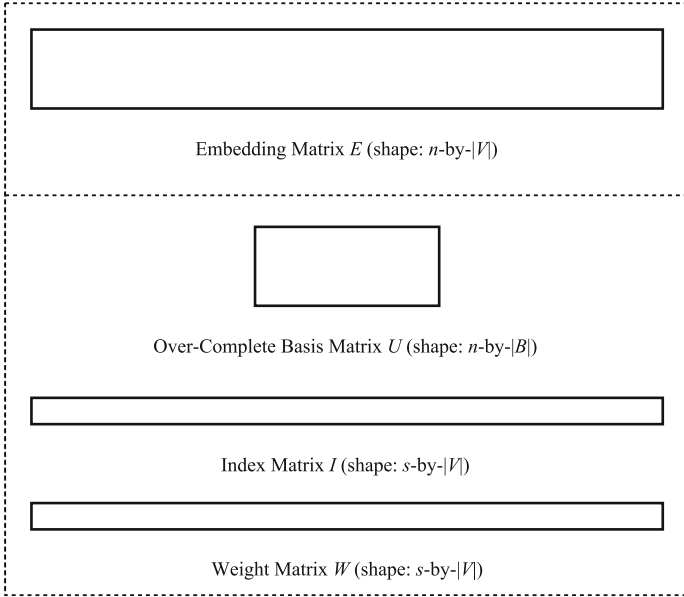


Fig. 1. Decompose embedding matrix into 3 smaller matrices

source of computation cost in RNN language models, and reduce the total computation.

Our goal is to calculate $P\mathbf{h}_t$ in Eq. 4 given a sparse decomposition of $P \in \mathbb{R}^{|V| \times n}$. Noting that the basis matrix is shared across all sparse representations, we can cache the product between basis matrix \hat{U} and hidden state \mathbf{h}_t based on the following key observation:

$$\langle P_i, \mathbf{h}_t \rangle = \left\langle \sum_{j=1}^s \text{weights}[j] \hat{U}_{\text{indices}[j]}, \mathbf{h}_t \right\rangle = \sum_{j=1}^s \text{weights}[j] \langle \hat{U}_{\text{indices}[j]}, \mathbf{h}_t \rangle, \quad (6)$$

where *weights* and *indices* are sparse representation for P_i .

Denote the sparse decomposition of $P \in \mathbb{R}^{|V| \times n}$ by $\hat{U}, \hat{I}, \hat{W}$ (they are of shape $|B|$ -by- n , $|V|$ -by- s , and $|V|$ -by- s respectively, not to be confused with input embedding matrix E 's decomposition U, I, W in Fig. 1), we have the following fast multiplication Algorithm 1.3.

Reduction in Flops: The original $P\mathbf{h}_t$ requires $n|V|$ float multiplications and $(n-1)|V|$ additions, while Algorithm 1.3 requires only $n|B|+s|V|$ multiplications and $(n-1)|B|+(s-1)|V|$ additions. Again, if $n = 400, s = 10, |B| = 2000$ and $|V| = 20000$, we reduce the computation cost by a factor of $(n|V| + (n-1)|V|)/(n|B| + s|V| + (n-1)|B| + (s-1)|V|) = 8.08$.

Algorithm 1.3. Fast Multiplication Algorithm

```

1: procedure FAST-MULTIPLICATION( $\hat{U}, \hat{I}, \hat{W}, \mathbf{h}_t$ )
     $\triangleright$  Compute  $P\mathbf{h}_t$  from a sparse decomposition of  $P$ , i.e.:  $\hat{U}, \hat{I}, \hat{W}$ 
2:    $\mathbf{v} \leftarrow \hat{U}\mathbf{h}_t$   $\triangleright \mathbf{v}$  is of length  $|B|$ 
3:    $\mathbf{r} \leftarrow \mathbf{0}_{|V|}$   $\triangleright \mathbf{r}$  is a zero-vector of length  $|V|$ 
4:   for  $i = 1..|V|$  do
5:      $r_i \leftarrow \sum_{j=1}^s \hat{W}_i[j]v_{\hat{I}_i[j]}$   $\triangleright \hat{W}_i, \hat{I}_i$  denote the  $i$ -th row of  $\hat{W}, \hat{I}$  respectively
6:   end for
7:   return  $\mathbf{r}$ 
8: end procedure

```

4 Experiments

In this section we show our experiment results on PTB dataset¹ and model performance on cellphones. To recap, our model is basically an RNN language model described in Subsect. 2.3, where the input embedding E_{w_t} in Eq. 3 is computed using Algorithm 1.2 and $P\mathbf{h}_t$ in Eq. 4 is computed with Algorithm 1.3. We use LSTM as the default RNN cells.

We pretrain input and output embeddings with skip-gram models using python package *gensim* [12] and draw 5 negative samples for each word in training data. The input and output embedding matrices in this paper corresponds to member variables *syn0* and *syn1neg* in class *gensim.models.KeyedVectors* respectively. Then we use Algorithm 1.1 to decompose pretrained embedding matrices and initialize parameters of our sparse RNN language model. For parameters within hidden layers, we simply initialize them from uniform distribution as in [2]. The index matrices I and \hat{I} are fixed thereafter, while basis matrices and weight matrices are kept finetuned during the training phase, which is another difference from paper [7]. Training is performed using *TensorFlow* [13].

4.1 PTB

For PTB dataset, we set up two experiments: small and large. All our hyperparameters and training protocols follow [2]². Both the small model and large model have a vocabulary size of 10,000 and 2 LSTM layers. However, the hidden size is different: 200 for small model and 1,500 for large model. The perplexity results are reported in Table 1 (the lower, the better):

We see that our sparse model has a higher training perplexity, but the gap between train and test perplexity is smaller. Actually, the sparse constraints act as a regularizer and prevent overfitting. We see that small sparse model performs better than the standard model, but large sparse model is worse. This phenomenon can be explained by the ratio $|B|/n$: for the small model, the bases

¹ Available at <http://www.fit.vutbr.cz/~imikolov/rnnlm/simple-examples.tgz>.

² See https://github.com/tensorflow/models/blob/master/tutorials/rnn/ptb/ptb_word_lm.py for details.

Table 1. Perplexity on PTB dataset

Model		Train	Test
Small	Standard [2]	37.99	115.91
	sparse ($s = 10, B = 2k$)	69.67	110.88
Large	Standard [2]	37.87	78.29
	sparse ($s = 20, B = 4k$)	55.23	82.35

are more over-complete ($|B|/n = 2000/200 = 10$), and the sparse approximation is pretty precise; while for the large one, the ratio is only $4000/1500 = 2.67$, which leads to some approximation error and causes degeneracy in performance.

4.2 Performance on Cellphones

In this part, we compare 3 different models. The first model is a standard RNN language model, with embedding size 400 and 2 LSTM hidden layers of size 400. The second one uses Algorithm 1.1 to compress the output embeddings and Algorithm 1.3 to speedup prediction. The third model further compresses the input embedding matrix.

The models are trained on an internal corpus, which is consisting of 10M sentences and the domain is daily conversation. We normalize all punctuation to <pun> and numbers to <num>, and keep the most frequent 20,000 words in the final vocabulary. For sparse representation, we set basis size $|B| = 2000$ and sparseness $s = 10$.

On a Macbook Pro Retina 2015, we test the memory consumption and response time (time for inference 1 step) of these 3 models, and summarize the results in Table 2. The response time is the average value of 200 inferences.

Table 2. Memory and response time on Macbook

Model	Model1: basic	Model2: sparse softmax	Model3: sparse
Memory consumption (MB)	72	47	24
Response time (ms)	16.5	9.5	7.5

From model 1 to model 3, we can see memory does reduce a lot due to our compression algorithm. And there is a large drop in response time from model 1 to model 2, which is the effect of our fast multiplication algorithm. Surprisingly, the response time of model 3 is even shorter than that of model 2, which can be attributed to better locality and modern cache system.

We also test our models on a Nexus 5, and we don't know any previous work that has tested their method on real cellphones. The memory consumption is roughly the same as that on the Macbook, so we omit it here. We run 100 predictions for each model, and list the response time as follows (Table 3):

Table 3. Response time on Nexus 5

Model	Model1: basic	Model2: sparse softmax	Model3: sparse
Response time (ms)	30~33	15~28	19~22

It’s clear that the last two models are faster, and the third model is more stable than the second one.

We also compare our model’s performance by combining it with LatinIME, an open source input method editor. The default language model of LatinIME is based on n-gram models, and it integrates unigram to trigram counts with a complicated algorithm and empirical values. When predicting next word, it utilizes previous context and current incomplete character sequence. Character sequence is fed to an internal Trie tree to find words with similar spellings, and the language model reranks the candidate words based on previous context. We replace the n-gram language model with our RNN language model, and compare the input efficiency of these two methods. **The input efficiency is defined as the ratio of number of real characters to that of keystrokes.** For example, if a user wants to input the word “happy”, and he managed to achieve this by typing only the first 3 letters “hap” (because input method recommend the word “happy” to him), the input efficiency is $\text{len}(\text{“happy”})/\text{len}(\text{“hap”}) = 5/3$. The test result is in Table 4.

Table 4. Input efficiency statistics

Method	LatinIME	LatinIME with RNNLM
Input efficiency	1.55	1.83

We see that LatinIME with RNNLM behaves significantly better than original LatinIME with n-gram language model. Actually, we do observe bad predictions of the original LatinIME like “in two days ago”, because it thinks both “in two days” and “two days ago” are valid. When combined with RNNLM, these bad cases rarely occur.

5 Future Works

In this paper, we propose a method to decompose a word embedding matrix into an over-complete basis matrix, an index matrix, and a weight matrix. By fixing the length of each sparse coding and computing the logits of next word distribution though the linear combination of those of output basis vectors, we reduce both the size and computation cost of the model and make it more suitable to run on hardware.

In the future, we plan to explore other kinds of redundancy, e.g.: sharing the input and output over-complete basis matrices, tying input and output embedding matrices, etc. These techniques will make the model smaller and more efficient.

References

1. Mikolov, T., Karafiát, M., Burget, L., Cernocký, J., Khudanpur, S.: Recurrent neural network based language model. In: Interspeech, vol. 2, p. 3 (2010)
2. Zaremba, W., Sutskever, I., Vinyals, O.: Recurrent neural network regularization. arXiv preprint [arXiv:1409.2329](https://arxiv.org/abs/1409.2329) (2014)
3. See, A., Luong, M.T., Manning, C.D.: Compression of neural machine translation models via pruning. In: Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, 11–12 August 2016, pp. 291–301 (2016)
4. Narang, S., Diamos, G., Sengupta, S., Elsen, E.: Exploring sparsity in recurrent neural networks. arXiv preprint [arXiv:1704.05119](https://arxiv.org/abs/1704.05119) (2017)
5. Chen, W., Wilson, J., Tyree, S., Weinberger, K., Chen, Y.: Compressing neural networks with the hashing trick. In: International Conference on Machine Learning, pp. 2285–2294 (2015)
6. Lu, Z., Sindhvani, V., Sainath, T.N.: Learning compact recurrent neural networks. In: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2016, Shanghai, China, 20–25 March 2016, pp. 5960–5964 (2016)
7. Chen, Y., Mou, L., Xu, Y., Li, G., Jin, Z.: Compressing neural language models by sparse word representations. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, Berlin, Germany, 7–12 August 2016, vol. 1: Long Papers (2016)
8. Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural Comput.* **9**(8), 1735–1780 (1997)
9. Gers, F.A., Schmidhuber, J., Cummins, F.: Learning to forget: continual prediction with lstm (1999)
10. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint [arXiv:1301.3781](https://arxiv.org/abs/1301.3781) (2013)
11. Press, O., Wolf, L.: Using the output embedding to improve language models. In: Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, 3–7 April 2017, vol. 2: Short Papers, pp. 157–163 (2017)
12. Řehůřek, R., Sojka, P.: Software framework for topic modelling with large corpora. In: Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks, pp. 45–50, Valletta, Malta, May 2010. ELRA (2010). <http://is.muni.cz/publication/884893/en>
13. Abadi, M., et al.: Tensorflow: a system for large-scale machine learning. In: OSDI, vol. 16, pp. 265–283 (2016)

Sentiment Analysis



Predicting Email Opens with Domain-Sensitive Affect Detection

Niyati Chhaya¹, Kokil Jaidka²(✉), and Rahul Wadbude³

¹ Big Data Experience Lab, Adobe Research, Bangalore, India

nchhaya@adobe.com

² University of Pennsylvania, Philadelphia, USA

jaidka@sas.upenn.edu

³ Indian Institute of Technology Kanpur, Kanpur, India

Abstract. The content and style of the text constitutes the semantic context of its words—a property that is important for many downstream tasks in natural language processing. We demonstrate the advantages of incorporating domain information for affect analysis, and subsequently for the prediction of user responses to marketing emails. Emails are a primary form of marketing communication, and email subject lines are the only indicators of whether the receiver will open an email especially in the case of bulk communication. We analyze the performance of affective features in predicting email opens, on a dataset of 60,000 unique promotion emails from 3 different industries. Our results show that the use of domain-specific affect words is strongly correlated with email opens and outperforms words from the standard ANEW lexicon and other state of the art affective lexica. Implications of this findings can be incorporated into writing tools to improve the productivity of marketing campaigns.

Keywords: Affect analysis · Email marketing · Linear programming · Convex optimization

1 Introduction

The email subject line plays a critical role in determining whether the email will be opened. It is the single point of insight regarding the email content for the recipient. This study presents a language-based model to predict the open rate of outbound marketing emails. Our experiments demonstrate the importance of linguistic features for this task. We also show how using domain-specific lexica, as against a standard affect lexicon (e.g. ANEW [1]) are able to tailor a generic predictive model to better predict the open rate for industry-specific emails. Our experiments highlight the word-usage preferences for different businesses and show how they vary across industries. Insights from this study can be applied by content writers to create improved email experiences as well as to better understand the psycholinguistic preferences of their customers.

The work makes the following contributions:

1. It implements a **framework to mine domain-specific affect lexica from any corpora of long or short texts**.
2. It demonstrates the **strong univariate relationships of the new lexica with open rates**, which outperform generic affective lexica.
3. It identifies the **word preferences that characterize high open rate for different industries**. Words providing insight and signaling cognitive processing lead to more opens for the Finance industry; on the other hand, social words yield more opens for the Movies & Television industry.

1.1 Paper Organization

Section 2 presents an overview of prior work in the space of email understanding and explorations with linguistic features. The method describing the regression model as well as the construction of domain-specific affect lexica is discussed in Sect. 3. An analysis on the linguistics features followed by the experiments is presented in Sect. 4. We conclude with a note on further explorations in Sect. 5.

2 Related Work

Early studies of users’ actions on emails were conducted in a relative small scope on a small set of monitored users [2]. Recent studies have looked at contact interactions [3], the effect of personalization [4] and the role of text-agnostic features [5] for predicting email opens; however, no study has attempted to predict email opens based on the sentiment in the subject line.

Our approach is based on building a custom domain-specific affect lexicon to model linguistic features for the prediction task. General-purpose affective lexica are used to detect emotions and sentiment at the word level, in various natural language tasks [1,6–8]. However, standard lexica often fail to capture the domain-specific orientation of the content. Several approaches have adapted general-purpose lexica for research problems in specific domains [9], [?]. We use an approach similar to studies which have explored the use of syntactic structure – such as parsing rules, linguistic patterns [9–11], and Latent Semantic Analysis [12–14] and collocations [15] – to identify domain-specific affect words. This is the first paper to apply the use of domain-specific lexica for **predicting email open rates**, since previous work has mostly focused on opinion mining tasks for product reviews.

3 Method

The first objective of this study was to extract different linguistic and meta-features from the labeled corpora, and build three industry-specific predictive models to predict the open rate for individual emails. The second objective is to understand the impact of affect words in open rate prediction.

3.1 Data Collection

Data access was provided by Edatasource¹, an email inbox monitoring organization which tracks over 25 million emails for 90,000 distinct businesses per day. The data is categorized into 98 industries. Using their licensed API, we were able to download the following information for up to 20,000 promotional emails each, sent over a one-year period (April 2015 to March 2016) and over fifty businesses each in Finance, Cosmetics, and Movies & Television:

- **Email information:** The subject line, contents, send date, time and time zone, sending email domain, name of the business, and industry category.
- **Recipient responses:** The number of the promotional emails which were received in tracked inboxes, percentage proportion of the recipients who read the email, and a projection of the total number of recipients for the message.

3.2 Domain-Sensitive Affect Detection – BATframe

We adapted an optimization-based approach to build domain-specific lexica for subject lines from three industries. This approach was proposed in [16] and outperformed the state of the art in the SemEval 2007 Affect Corpus, with a precision of over 70% as compared to the best performing system at 47%. (Fig. 1).

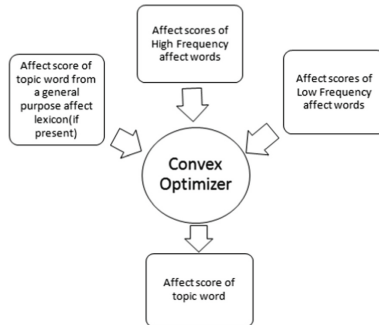


Fig. 1. The BATFrame Framework for Domain-sensitive affect detection [16].

According to this approach the affinity between affect words in the neighborhood of topic (domain) words is modeled as a optimization function in this approach. First, the subject lines are tokenized using HappierFunTokenizer² to produce a total of 0.9 million tokens. Next, the Topic-affect Tuple Extractor word pairs that couple the topic space with standard affect words, using n-grams and dependency rules. Finally, the optimization Framework tunes the

¹ See <http://www.edatasource.com>. Edatasource monitors the email inboxes of millions of email users, after obtaining their consent, and saves email contents and user responses in a de-identified form for the purposes of marketing research.

² <http://sentiment.christopherpotts.net/>.

domain-specific Pleasure (Valence), Arousal, and Dominance (P,A,D) scores for the topic word on the basis of a set of constraints. The mathematical expression is as follows:

$$\begin{aligned} \omega_{PAD} = & \lambda_1 \sum_{j=1}^n I_{w_j}^G \|S_{w_j} - G_{w_j}\|_2 \\ & + \lambda_2 \sum_{j=1}^n \sum_{a_k \in HF_j} \alpha_{jk} \|S_{w_j} - G_{a_k}\|_2 \\ & + \lambda_3 \sum_{j=1}^n \sum_{b_k \in LF_j} \alpha_{jk} \|S_{w_j} - G_{b_k}\|_2 \end{aligned} \quad (1)$$

Now the optimization problem is given by $S_p = \min \omega_{PAD}$, subject to:

$$1 \leq S_{jp} \leq 9 ; 1 \leq S_{ja} \leq 9 ; 1 \leq S_{jd} \leq 9 \quad (2)$$

where λ_1, λ_2 are weighting parameters which should be set to the degree that we trust each source of information, and λ_3 can be set to a small non-zero value such as 0.002. Table 1 illustrates some resulting domain-specific topic words from the three corpora, that were not present in ANEW in any lemma form, which demonstrates how domain-specific lexica capture more affective content as against standard lexica.

Table 1. The top ten highest-weighted domain-specific words in the BATFrame lexica which are not present in ANEW or in Warriner’s Lexicon.

Corpus	New Affect words	New N
Finance	app, dividend, anyone, discount, quantitative, +, data, divergence, flight, authentication	198
Cosmetics	men, addition, flirtiest, everyone, off, you, more, 5-star, matte, own	73
Movies & TV	today, prime-time, easy-to-please, nail-art, one-pot, well, loud, while, %, front	100

4 Experiments

The purpose of this evaluation is to test whether domain-specific lexica contributes to predicting email opens. We posit that by producing new features along the three dimensions of Pleasure, Arousal and Dominance, the BAT lexicon will improve on the performance over standard lexica to predict email opens. We conducted univariate regression analysis to predict the open rate of emails using (i) meta-features and POS tags and (ii) the three dimensions of affect from ANEW [1], the Warriner’s lexicon [6], and the BAT lexica. In order to do so, we generated features representing different kinds of meta-information (subject

line length and word count), the percentage use of punctuations and symbols, and part-of-speech tags in each subject line using the TweetNLP tagger, which is trained on social media text [17]. Figure 2 depicts the 1-to-3 g positively and negatively correlated with open rate. All the correlations were Bonferroni-corrected, and are significant at $p < 0.01$. The size of the word reflects its correlation with open rate, while the shade reflects its frequency in the dataset. In Cosmetics, words such as ‘registration’ and ‘member’ and phrases such as ‘welcome to’ led to more opens; on the other hand, phrases mentioning ‘notifications’ and discounts (‘%’) were negatively correlated with opens. In Movies & Television, subject lines with ‘you’ and phrases such as ‘is now’ were more likely to be opened, and subject lines mentioning news coverage (‘breaking news’) were less likely to be opened.

Table 2. Standardized regression coefficients (β s) between different features of subject lines, and email opens. Subject lines containing the positively correlated features below are significantly more likely to be opened. All correlations are significant at $p < .01$, two tailed t-test.

Finance		Cosmetics		Movies & Television	
Feature set	R^{**}	Feature set	R^{**}	Feature set	R^{**}
Meta-features & parts of speech					
Word count	-.11	Brackets	.17	Possessive pronoun	.20
Verb	-.10	Proper noun	.09	Verb, third person singular	.16
Currency	-.08	Present tense	-.08	Numbers	-.12
All punctuations	-.11	Quantity	-.05	:	-.14
ANEW					
Arousal	.06	Arousal	-.06	Arousal	.05
Valence	-.05	Valence	-.10	Valence	.04
Dominance	-.10	Dominance	-.09	Dominance	-.06
Warriner’s lexicon (Extended ANEW)					
Arousal	.10	Arousal	.05	Arousal	.03
Valence	.09	Valence	-	Valence	.07
Dominance	.12	Dominance	.04	Dominance	.08
BATFrame					
Arousal	.11	Arousal	-.14	Arousal	.10
Valence	.10	Valence	-.14	Valence	.09
Dominance	.09	Dominance	-.15	Dominance	.11

Table 2 illustrates the text features among ANEW features, Warriner’s (extended ANEW) features, BAT lexica features, meta-features and parts of speech, which were most highly correlated with Open Rates for the three

industries. The effect sizes for individual features ranged from -0.18 to 0.23 across the industries.

The table enable us to compare the characteristics of subject lines across various industries. We observe that for different industries, different words, phrases, and topics are more likely to yield higher open rates:

- **Meta-features and POS:** Short and crisp subject lines devoid of punctuation are evidently preferred in the Finance industry, and are more likely to be opened; on the other hand, proper nouns perform well in Cosmetics, and possessive pronouns do well in Movies & Television.
- **ANEW, Warriner’s Lexicon and BATFrame:** BATFrame outperforms ANEW and also Warriner’s Lexicon in all three corpora. ANEW has the poorest performance with the weakest coefficients. Warriner’s lexicon is comparable to BATFrame in the Finance corpora.
- **BATFrame features:** These features highlight that while Valence and Arousal features have similar importance across industries, the importance of Dominance terms is varies.

We also trained three multivariate linear regression models to predict email open rates on a held out test set. The feature set comprised standard tf-idf features [18] calculated from the n-gram distributions complemented with one of the three affective lexica. Because of the large effect size and number of features available from words, there was no significant difference in the effect sizes from either the ANEW, the Warriner’s, or the BAT lexica, and all three models yielded an average Mean Absolute Error of 0.07 across the three corpora.

4.1 Qualitative Evaluation

Figure 3 shows that the domain-specific BAT lexica offer more consistent coverage in corpora from all three industries, after excluding stop words. Coverage can be interpreted as how representative any lexicon is of the overall vocabulary of the test set. The BAT lexica achieves around 85% coverage of the vocabulary corpus as against the 10% coverage by ANEW. BAT-Finance lexica outperforms Warriner’s Lexicon (approx 13k tokens) as well which is 14 times the size of the BAT-Finance (975 tokens). Note that the BAT lexica (BAT-cosmetic:419, BAT-TV:848) are significantly smaller in size as compared the standard lexica (ANEW:1034).

The results highlight the importance of a domain-dependent lexicon for the accurate affect analysis of short texts. This supports our argument that a domain-specific lexicon would be more representative of text than a general-purpose lexicon.

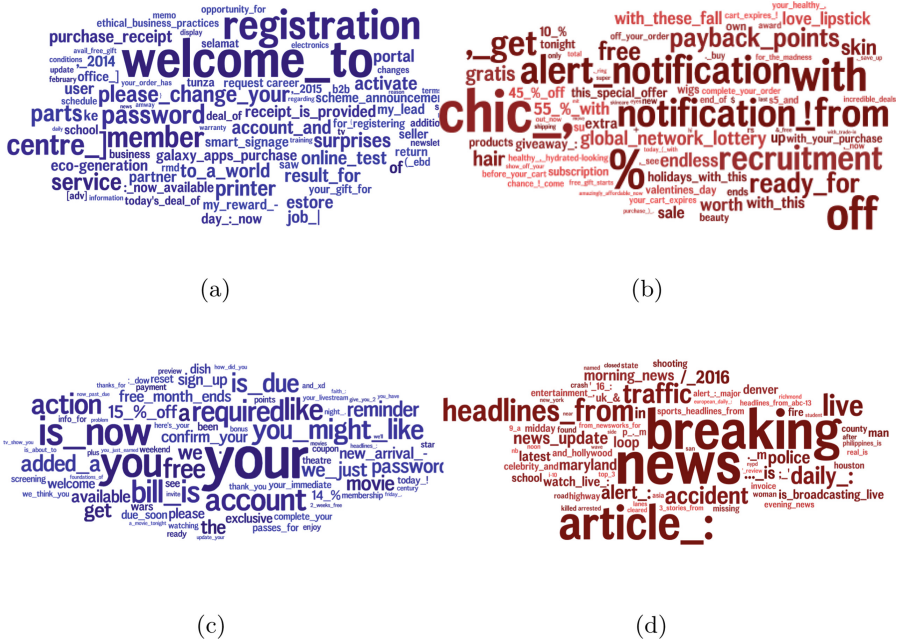


Fig. 2. The word cloud shows the ngrams that are significantly correlated with the open rate in the Cosmetics (a,b) and Movies & Television (c,d) industry. The word clouds in blue and red represent the positively and negatively correlated words respectively. The size of the ngram is proportional to its correlation with the open rate. The shade of the color signifies the frequency of occurrence; darker shades imply higher frequency as compared to lighter shades. (Color figure online)

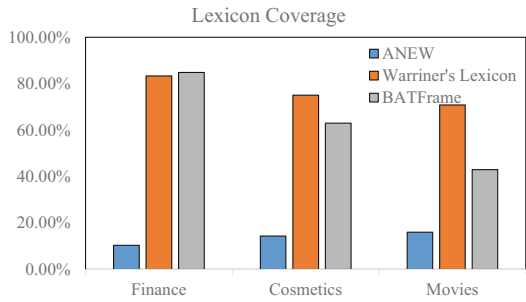


Fig. 3. Coverage Statistics on the Finance, Cosmetics and Movies & Television corpora. This figure shows that domain-specific lexica generated from BAT a significant amount of total vocabulary of a corpus, which is comparable to the Warriner's lexicon, and four times as much as ANEW.

5 Conclusion

Our study demonstrates that domain-specific affect lexica can improve sentiment and affect detection in different applications and for several predictive problems. We establish the importance of affect-based linguistic features for email analytics on a real-world dataset and further contrast the language preferences across data three industries: Finance, Cosmetics, and Movies & TV. We are currently extending this work towards generating suggestions for improved email opens. Our results also suggest that such approaches could be useful for word sense disambiguation.

References

1. Bradley, M.M., Lang, P.J.: Affective norms for english words (anew): instruction manual and affective ratings. Technical report, Technical report C-1, the center for research in psychophysiology, University of Florida (1999)
2. Lim, K.H., Lim, E.P., Jiang, B., Achananuparp, P.: Using online controlled experiments to examine authority effects on user behavior in email campaigns. In: Proceedings of the 27th ACM Conference on Hypertext and Social Media, pp. 255–260. ACM (2016)
3. Di Castro, D., Karnin, Z., Lewin-Eytan, L., Maarek, Y.: You’ve got mail, and here is what you could do with it!: analyzing and predicting actions on email messages. In: Proceedings of the Ninth ACM International Conference on Web Search and Data Mining, pp. 307–316. ACM (2016)
4. Sahni, N.S., Wheeler, S.C., Chintagunta, P.K.: Personalization in email marketing: the role of non-informative advertising content (2016)
5. Luo, X., Nadanasabapathy, R., Zincir-Heywood, A.N., Gallant, K., Peduruge, J.: Predictive analysis on tracking emails for targeted marketing. In: Japkowicz, N., Matwin, S. (eds.) DS 2015. LNCS (LNAI), vol. 9356, pp. 116–130. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24282-8_11
6. Warriner, A.B., Kuperman, V., Brysbaert, M.: Norms of valence, arousal, and dominance for 13,915 English lemmas. *Behav. Res. Methods* **45**, 1191–1207 (2013)
7. Taboada, M., Brooke, J., Tofiloski, M., Voll, K., Stede, M.: Lexicon-based methods for sentiment analysis. *Comput. Linguist.* **37**, 267–307 (2011)
8. Esuli, A., Sebastiani, F.: Sentiwordnet: a publicly available lexical resource for opinion mining. In: Proceedings of LREC, vol. 6, pp. 417–422. Citeseer (2006)
9. Qiu, G., Liu, B., Bu, J., Chen, C.: Opinion word expansion and target extraction through double propagation. *Comput. Linguist.* **37**, 9–27 (2011)
10. Blitzer, J., Dredze, M., Pereira, F., et al.: Biographies, bollywood, boom-boxes and blenders: domain adaptation for sentiment classification. In: ACL, vol. 7, pp. 440–447 (2007)
11. Chikersal, P., Poria, S., Cambria, E., Gelbukh, A., Siong, C.E.: Modelling public sentiment in twitter: using linguistic patterns to enhance supervised learning. In: Gelbukh, A. (ed.) CICLing 2015. LNCS, vol. 9042, pp. 49–65. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-18117-2_4
12. Turney, P.D., Littman, M.L.: Measuring praise and criticism: inference of semantic orientation from association. *ACM Trans. Inf. Syst. (TOIS)* **21**, 315–346 (2003)
13. Bestgen, Y.: Building affective lexicons from specific corpora for automatic sentiment analysis. In: LREC (2008)

14. Bestgen, Y., Vincze, N.: Checking and bootstrapping lexical norms by means of word similarity indexes. *Behav. Res. Methods* **44**(4), 998–1006 (2012). <https://doi.org/10.3758/s13428-012-0195-z>
15. Wiebe, J., Wilson, T., Bell, M.: Identifying collocations for recognizing opinions. In: *Proceedings of the ACL-01 Workshop on Collocation: Computational Extraction, Analysis, and Exploitation*, pp. 24–31 (2001)
16. Jaidka, K., Chhaya, N., Wadbude, R., Kedia, S., Nallagatla, M.: BATframe: an unsupervised approach for domain-sensitive affect detection. In: Gelbukh, A. (ed.) *CICLing 2017*. LNCS, vol. 10762, pp. 20–34. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-77116-8_2
17. Owoputi, O., O'Connor, B., Dyer, C., Gimpel, K., Schneider, N., Smith, N.A.: Improved part-of-speech tagging for online conversational text with word clusters. *Association for Computational Linguistics* (2013)
18. Ramos, J., et al.: Using tf-idf to determine word relevance in document queries. In: *Proceedings of the First Instructional Conference on Machine Learning*, vol. 242, pp. 133–142 (2003)



Detection of Suicidal Intentions of Tunisians via Facebook

Jihen Nasri^(✉) and Salma Jamoussi

University of Sfax Higher Institute of Computer Science and Multimedia of Sfax, B.P.: 242,
3021 Sfax, Tunisia
jihene.nassri@gmail.com
<https://www.isimsf.rnu.tn/>

Abstract. As response to the urgent problem of the continuous suicidal rates increases in Tunisia since the revolution of January 2011, this paper suggests the detection of suicidal intentions of Tunisians through Facebook. Indeed, among our major contributions we can cite: First, the usage of the Facebook social network due to its popularity in Tunisia required the development of a Facebook application to extract data, we also added some components to achieve our goal and form our corpus for natural language processing. In the second place, the identification of the best method of classification and the set of the most relevant attributes in identification of suicidal ideation especially among Tunisians. We reached encouraging results after a number of classification experiments thanks to an empirical comparison of the obtained performances by several subsets of features.

Keywords: Social networks · Data mining · LIWC · Facebook application

1 Introduction

According to the World Health Organization (WHO), yearly, almost one million people die by committing suicide. WHO, has declared: “It is one of the leading causes of death in 15-44 years old people. In the last 45 years, suicide rates have increased by 60% in some countries” [1]. Otherwise, locally we witnessed that a suicide case, which is the immolation of Bouazizi was the trigger of the Tunisian revolution on the 14th January 2011. Considering this move as heroic way to rebel against corruption and injustice, Tunisians took the suicide act to the next level. Since 2011, the suicide rate in Tunisia has risen up continuously due to the contagious effect of the Bouazizi. Facing such emergency efforts have been dedicated to screening individuals with suicide intentions to save them before they attempt suicide or harm themselves. However, this process is challenging in two broad ways : Detecting suicide ideation is in itself a defiant step to do, but also encouraging young people with a low immune background culture to assist to awareness campaign and consult psychiatrists is not easy.

Crossing through all those confrontations, we investigated the potential of social media mining as a method for suicidal intention detection. Regardless of their position social media platforms enable the rapid information exchange between users. “As of

2017, daily social media usage of global internet users amounted to 135 min per day” [2], based on this statistics imagine that we multiply this 135 minutes per social media users 2.46 billions for 2017 [3] and per 365 days. Can you imagine how huge the volume of insightful knowledge in such rich web resources can be in just one year? Accordingly, monitoring social media networks enables to capture the potential suicidal users and address them to the needed resources, even without leaving their homes. Big data analytics techniques can support this goal, by uncovering early signs of suicide ideation. Referring to the literature, data mining can be divided into two major portals of research which are the descriptive approach and the predictive approach. The descriptive approach relies on the lexicon based methods while the predictive approach make use of the machine learning methods. Indeed, machine learning aims at building models inferring what is happening behind some data so that it can predict future outcomes concerning the defined purpose. Our purpose is detection of suicide intention from social media content and taking in consideration that is a predictive task we rather to rely on the machine learning methods. In the Tunisian case, statistics show that most of Tunisians tend to be Facebook users, 71,25% of Tunisians [4]. On this wise, we propose to work on Facebook sites where we can target countless suicidal Tunisian cases by building a new predictive classification model able to detect suicidal Tunisian Facebook users.

In this paper, we propose to work on satisfying the objective of detecting suicide intentions for Tunisian people via Facebook. Although works were extending very far on suicide prediction via social media, none of the previous researches worked with Facebook sites nor focused on Tunisian people. Accordingly, in this work we were interested in solving the problem of defining the relevant features that might help in the suicidal ideation identification particularly for Tunisian people. Our first contribution was building the first corpus containing suicidal and non suicidal Tunisian people profiles features where we found an appropriate way to derive the data from the Facebook site by the mean of developing a data retrieval tool of Tunisian dialect status from Facebook. Our second contribution was defining a relevant set features of suicide for Tunisian people where we mixed both social media and personality features. On the basis of those features we obtained an effective classification method which predict the suicidal intentions of the Tunisians Facebook users. Related to our contributions, our research aims to answer the two following questions: Is our data retrieval tool effective in extracting real suicide data? What is the most accurate classification method based on our corpus?

2 Related Work

Digging into the literature from both fields data processing and psychiatry, we find a match between researches from both fields on the basis of suicide ideation motives. Accordingly, we select two major motives: Psychological disorder and to be exposed to discussion which implies suicidal content.

Psychological Disorder. Depression or psychological distress has been linked to different personality traits and behaviors including suicidal behaviors ([5, 6] and [7]). “Extended feelings of depression in a young adult may create a lack of purpose or meaning

in life, which, in turn, may result in a preoccupation with thoughts of suicide.” [8]. Thus, works were generated about depression detection via social media such as (Munmun, Michael, Scott & Eric, 2013) [9] who have developed MDD¹ classifier that predicts if the individual exposed to depression or not with an obtained accuracy of 70% using Support Vector Machine (SVM) with RBF kernel as classification technique. Several important features were introduced and divided into six categories which are engagement, egocentric social graph, emotions, linguistic style, and depressive sentiment vocabulary.

In psychiatry field, a study concluded that “Traumatic grief was associated with a 5.08 times greater likelihood of suicidal ideation, after control for depression” [10]. In this context, [11] were interested to derive grief and emotion distress from the messages posted to the profile of deceased MySpace users. Therefore, they put into service MyDeathSpace² to identify the deceased MySpace users then they pick comments expressing emotional distress based on 6 rules of codebook. Instead of employing the qualitative approach (content and thematic analysis) as in the previous studies they focused on quantitative analysis of the content and linguistic style of post mortem comments in order to understand the post-mortem interaction in social media. They used many features such as linguistic style and sentiment expression (positive emotions, sadness, anger, social processes, social relationship). This work come up with two binary logistic regression models that are strong predictors for the detection of emotional distress in a given comment.

People may communicate suicide content due to depression state or grief of losing someone by sharing text on social media but the real problem is to distinguish between worrying languages about real suicide intention and flippant references to suicide, [12] were interested to solve this issue by the mean of classification methods of SVM, Decision Tree and Naive Bayes. To perform the classification they relied on two categories of features which are lexical characteristics and sentiment characteristics.

Suicidal Discussion Impact. One evidence is that “everyone is a product of their environment”, social media sites already constructed an environment for it users. In this context, we should investigate the suicidal content effect from the possibility of it propagation until it impact on who interferes in suicide discussions.

Thus, [13] focused on the connectivity and communication of suicidal ideation between social media user. This work relied on two type of features sets which are connectivity described by friends and followers distribution graph and the communication described by retweet graph. In friends and followers distribution graph the nodes present suicidal users and edges present links between those users (follow, friendship, mutual). In the retweet graph, the nodes present users who replied (i.e retweet) the suicidal posts and edges present 'has retweeted'. They performed graph theoretic where relevant metrics such as (nodes/edges number, density, transitivity, etc.) lead to the evidence of the contagious effect, same result as psychology field. This finding may lead to the spread of suicide-promoting.

¹ MDD: mental illness Major Depressive Disorder.

² MyDeathSpace: <http://mydeathspace.com>.

Some researchers focus on the discussion level and how it can be linked to suicide ideation shifts. For That [14] made an interesting study on forecasting if people engaged in mental health discussion would discuss suicide ideation. Their purpose was to prove that the link between the history of mental illness and future suicide risk can be important. Accordingly, they picked Reddit as appropriate social media network where posts are organized by areas of interest called “subreddits”. This work obtained post and comment data from mental health subreddits and subreddits about those contemplating suicide. Besides the linguistic features, they add other features sets which are interpersonal awareness(i.e proportion of first personal, proportion of singular, proportion of first personal plural, etc.) and interaction (Volume of posts and comments received /authored, vote differences, response velocity , etc.).

In the same context and same year, another group of researcher [15] take into consideration discussion level as feature by studying it impact on suicide detection. Their work proved that combining the stylistic linguistic and the discussion level feature improves the performance compared to the previous published results.

Otherwise, those related works basically relied on all social media networks sites except the Facebook in spite of it credibility and being the largest social network site. Also the proposed solutions worked only on the English language while suicide issue is global issue and the potential suicidal will express them selves in multilingual manners. Moreover, the usage of crowd sourcing in suicidal cases validation doesn't seem as a rigid technique to decide if the content on which we will perform training is really suicidal. In our case we choose to work on Tunisians therefore we will construct our corpus based on hybrid approach and we will treat the Multilanguage issue. Besides, We will work on Facebook as a popular and large social network site.

3 Proposed Method

3.1 Corpus Preparation

As working on Tunisian people was our contribution, no appropriate dataset was available. That's why creating our own corpus was our priority. Our interest was to fetch Facebook profiles of Tunisian people who deceased by committing suicide or who attempt suicide at least once, also we picked people who posted suicidal content regardless of their age, gender or their personal condition (job, status, location, etc). Our choice of diversified data aims to cover all Tunisians possible profiles as well as creating a reasonable corpus.

We launch investigations about suicide cases by visiting the online newspapers such as “JawharaFM”, “Kapitalis”, or consulting some TV shows about suicide cases such as “Tunisian Tales”, “None Lasting Status”. Starting from one photo and a name, we followed the same process as to find friend on Facebook site. Once we reach the Facebook profile target, we need to be certain and sure about the real identity of the Facebook user. The validation was based on the context of his Facebook friends posting: Only if their comments were about grief or a suicide attempt. Also, we give an interest to users sharing suicidal content even if they didn't commit suicide. In order to collect those profiles we put “#” + “key word” as a request in the Facebook search bar. Those key words were about suicide lexicon in Tunisian language. As result, the Facebook site displays a page where

posts containing those key words are listed including their authors. While we picked some of those Facebook users, we needed experts validation to add them as suicidal subjects. For that, we prepared a survey in a form of a table composed of 12 Facebook user profiles textual posts. To fill this survey we asked five different psychiatrists from both public and private sectors to collaborate with us by deciding if the subject has intentions of suicide or not. Once, we collected the five experts classification results, we apply voting strategy for the final classification of the 12 users.

3.2 Data Extraction

In the scientific community, the purpose of using data is to understand a particular phenomenon, perform analyzes or predict the future. In our case we need to understand the suicide phenomenon by performing analysis on the created corpus in order to predict suicide intention in the future. In our case, we can summarize this process into three steps: First, we need to collect the raw data (posts or personal informations) from each collected Facebook profile. Second, we should store the data. Finally, we must structure and prepare the data before any other treatment.

From the starting, the initial step of data extraction was an overwhelming issue. In spite of admitting that Facebook provides a graph API to derive data automatically, the access to another Facebook user data is constrained. In other words, you cannot get a subset of data stored on the Facebook of a particular Facebook user unless this user gives you his permission. In this context, how can we ask permission from died people by suicide in some collected Tunisian cases?

To answer this question, we had no choice but following the old school manner of data extraction, by the way of copy and paste the posts of each deceased Facebook user. As performing data retrieval manually was a time and effort consuming task, we come up with a solution which was only useful if we eliminated the deceased cases. The key idea was to develop a Facebook application in disguise of a quiz called “Cop Quizzer”, then we send the link to the selected Facebook profiles and ask them to click on “connect with Facebook” button. Any Facebook application is working with the Facebook Graph API, which is based on requesting an access token to get different permission such as asking for gender, birthday, posts, etc. From an implementation perspective, having a button “continue with Facebook” is equivalent as having an access token. In other words the developed Facebook application will out- put the same results as the Facebook Graph API. In our case, once the Facebook user click on “continue with Facebook” button action, a pop up page appears asking the user for his permission to access his personal data. If the user agrees, his latest 25 posts and his personal information (i.e birthday date, gender, status, etc) will be sent automatically in real time to our online database deployed on the cloud Firebase, else he will quit the quiz. Finally, Firebase provides us with the option to export the data in the form of a JSON file. Many efforts have been dedicated in data collection until we achieved a total number of 93 Facebook profiles and 2325 textual posts, where our concern was to keep the criteria “Tunisian”. Thus, we didn’t put any other conditions in selecting profiles as we strive for data diversity.

3.3 Data Processing

Data structuring task is not only limited on treating the emoticons nor correcting syntax errors but also handling the Tunisian dialect used language. For data structuring, we first integrated an algorithm for text normalization (emojis, emoticons, URLs, etc.). In addition, as we aim to solve the translation problem we relied on both machine and human translators.

Concerning the text normalization, our key idea is to replace the existing emoticons and Emojis in the derived text respectively by the suitable sentences based on their meaning instead of their elimination. In fact, we wanted to add the useful information that may be expressed by the emoticons and Emojis available in the textual content of our data.

Machine translation (i.e. computer-generated translation) alone cannot handle the problem of multilingual textual data. Thus, the contribution of human experts in the translation task is required in order to ensure correctly the data translation. For that, we assigned the task of translation for 7 English teachers who accepted to help us in this mission. As we want to avoid the effort and the time that may occur if we rely totally on the human translators we relied on Yandex.Translate free API. It's a Statistical Machine Translation (SMT) model. Yandex.Translate was developed in 2011 by Russian technology company. Today, it affords the translation of whole sentences, words or web pages between 93 languages.

3.4 Features Extraction

The added value of our work at this point, is the fact that we vary as much as possible in features. Which we may divide generally into personality features and social media features. The sum of those two categories is counting 64 features. Therefore it is recommending to mention that the personality features and social media feature can be complimentary as both constitute one person identity. Thus, if we reach to correctly define a person we may achieve to distinguish its suicidal aspect among those multiple personalities.

Personality Features. Based on the ability to link the daily usage of words to the real behavior, researchers came up with Linguistic Inquiry Word Count (LIWC), a powerful tool allowing to perform analysis on one individuals' words to reveal diversified traits of his personality. At the beginning LIWC aimed to analyze both the psychological processes and what people were writing or speaking about. Later after a sum of improvement, Receptivity API was built as tool linked to the popular psycho- linguistic analysis LIWC. In our work we implemented Receptivity API to calculate features automatically extracted from our created and refined Corpus (basically the textual posts). The output of the Receptivity API is a JSON-based result containing a set of 60 features calculated in percentile. These features belong to many possible categories among which we cite: *Emotionality*. Designed by cheerful, stressed, anxious, melancholy, insecure, neuroticism, happiness, depression and cold. These features means the degree expressed by people toward an entity which can be an event, object or person. From predictive perspective, emotions are the reflection of one's experience in this world. By the mean of

emotions, individual response and react to traumatic events, such reactions can tell us about how the concerned subject will deal with events in the future. *Social relationships*. Designed by openness, sociable, friendly, cooperative, impulsive, social skills, independent, adjustment and extraversion. According to [17] pronouns reveals referencing individuals interactions, word count shows the degree of dominance in conversation and positive emotion words declare the level of agreements. These features allow to track the social processes by determining who has the maximum of status, if the individual tend to be in group or in isolation. From personality perspective social relationships shows the engagement of the subject and his dominance in interaction with others. *Attentional focus*. Designed by Family oriented, friendship focus, body focus, health oriented, sexually focused, food focused, leisure oriented, money oriented, religion oriented and work oriented. Those features declare the individual interests and it priorities. Such findings can help to predict the behavior, for example if a child is playing football and he gets injured he won't notice and won't stop the game because he is fully focused on the game else is this child is body focused he will be worried, stop the game and go ask his parents help in tears. Thus determining ones priorities can help us deeply understand how people may act according to an event or situation. *Honesty and deception*. Designed by trusting, genuine, persuasive and imaginative. Those features works on identifying if the truthful statements from the deceiving ones. According to [17] fake statements rely on changing languages, too much of descriptive lexicon in order to convince the listener. Also the level of intelligence encounters as the naive persons tend to be more truthful and honest than smart people as imagination and cognitive are required to load and maintain a story which is the opposite of the true one where there is efforts to convince someone that the fake version of story is true.

Social Media Features. Among features than can be offered via social media derived by our Facebook application we were only concerned with: *Age*. Designs how old the subject, calculated on the basis of date of birthday and presented as integer. Age can play an important factor as level of maturity of one person is based on his age where teenagers tend to be more exposed to depression which can be a major motive of suicide. *Gender*. If the subject is male or female, this feature is presented as binary (i.e Male = 0, Female = 1). We picked gender as an important feature because studies show that suicidal behavior differs between male and female, suicide rating shows that females tend to have more suicidal thoughts however males pass directly into the action faster than female. *Marital status*. If the subject is in relationship or not by reference to his status and represented as binary (i.e Single = 0, In relation = 1). We thought about the status of the person, we supposed that if the person is committed to a relationship he will be more committed to life and he won't probably think about suicide. *Job status*. If the subject has job or not and represented as binary (i.e no job = 0, with job = 1). Job status is interesting as predictor, the proof is the immolation of Bouazizi in Tunisia who suffer from unemployment and his impact on the others who shared the same problem of joblessness, after his act they started to imitate him as if he was a role model. *Student status*. If the subject is perusing his studies or not and represented as binary (i.e not student = 0, is student = 1). We suggest to add this feature to see if the Tunisian students tend to be optimistic toward future and keep working to concur their desired future life which shows their attachment to life or to see if students are impacted by the negativity advertised by some broadcast

channels on the social networks or on the television and radios toward the obscure future in Tunisia that lead to describing the hopelessness until the desire to quit and give up life.

4 Experimental Results and Discussion

Using the WEKA software we tested a series of methods and algorithms to achieve the classification task and we pick up the obtained results to choose the most appropriate methods to apply on our case. In fact, a baseline classification will tell if we retained a rigid basis for modeling our prediction problem. Then, WEKA provide us with helpful feature selection methods to work on improving the results returned by the baseline classification.

From judgmental perspective, we picked in the following experiments three measures of evaluation which are accuracy, F-measure, and suicidal recall. After each classification task, WEKA returns the values of those evaluation measures indicating the performance level of our method.

4.1 Baseline Experiments

A baseline classification was applied based on all 65 features (Facebook & personality) with K-fold cross-validation as a technique of dividing data into training and test sets where we choose K= 10. Each time we pick a classifier, WEKA returns its appropriate calculated evaluation measures. On the purpose of comparison between the used machine learning classification methods we traced a Table 1 gathering the generated evaluation measures (suicidal Recall, Average F-measure and accuracy).

Table 1. Results of baseline classification based on all features

Method	Suicidal recall	F- measure	Accuracy %
Tree J48	0,78	0,77	77,41
Tree LMT	0,67	0,67	67,74
Adaboost M1	0,63	0,72	73,11
Logitboost	0,67	0,66	66,66
SVM (RBF kenel)	0,63	0,65	65,59
SVM (Poly Kernel)	0,71	0,72	72,04
Multilayer Perceptron	0,73	0,7	70,96
Naive Bayes	0,69	0,67	67,74

Referring to the Table 1, we can take evidence that Tree J8 is on the top of other classifiers with a percentile of 77,41% followed by Adaboost whose accuracy is 73,11%. Regarding the suicidal recall, Tree J48 has a distinctive result that is equals to 0,78

compared to the other classifiers. Investigating the relevance of each type of the two features sets (Facebook features and Personality features), a baseline classification was applied separately based on Facebook features set then on personality feature set with similar settings (i.e K=10 and same classifiers) as the previous experiments. On the purpose of comparison between the features sets (Facebook versus Personality) we traced a Table 2 gathering the generated evaluation measures (suicidal Recall, Average F-measure and accuracy). F refers to Facebook features and P refers to Personality features.

Table 2. Results of baseline classification based on separated features (Facebook versus Personality).

Method	Suicidal recall		AVG F-measure		Accuracy %	
	FB	P	FB	P	FB	P
Tree J48	0,80	0,58	0,83	0,6	83,87	60,21
Tree LMT	0,78	0,54	0,81	0,57	81,72	56,98
Adaboost M1	0,69	0,58	0,73	0,64	73,11	64,51
Logitboost	0,8	0,58	0,79	0,61	79,56	61,29
SVM (RBF kernel)	0,56	0,71	0,68	0,63	68,81	63,44
SVM (poly Kernel)	0,58	0,56	0,68	0,57	68,81	56,98
Multilayer Perceptron	0,71	0,56	0,78	0,57	78,49	56,98
Naive Bayes	0,63	0,6	0,72	0,55	73,11	60,21

With respect to the Table 2, we find that classification based on the Facebook feature set is more accurate than the one based on the personality features. For the same classifier the accuracy concerning FB features is always superior than accuracy concerning personality features except for the SVM classifiers (with RBF kernel and RBF kernel). The same remark can be also made for the suicidal recall values where the obtained scores of the FB features are more important than the ones obtained with Personality features excluding SVM classifiers.

4.2 Feature Selection Experiments

Working on improving the results returned by the baseline classification (see Table 1) WEKA provides us with helpful feature selection methods. Particularly, the relevance of chosen features leads to more accurate classification results. For that, we chose to work with Wrappers as feature selection method. The Wrapper is tied to the performance of specific classification model. For feature selection we used a number of classification methods (ie. Tree j48, SVM, Adaboost, etc) as Wrapper methods. Each experiments outputs a subset of random features which is reused as input in the classification task. On the purpose of comparison between the best results returned based on feature selection by wrappers,

we traced a synthesis Table 3 gathering the generated evaluation measures (suicidal Recall, Average F-measure and accuracy) and the features selected by each wrapper.

Table 3. Synthesis of the best classification results after attribute selection.

Wrapper	Selected features	Best classifiers	Suicidal recall	F-measure	Accuracy %	Rank
Logitboost	FB: age, gender, job status.	Tree J48	0,91	0,92	92,47	1
	P: melancholy, leisure oriented.	Tree LMT	0,82	0,84	84,94	6
		MLP	0,8	0,76	76,34	13
Multilayer Perceptron MLP	FB: gender, job status.	Tree J48	0,78	0,84	84,94	8
	P: friendly, humble, thinking style, family oriented, intellectual, liberal.	Adaboost	0,8	0,79	79,56	10
		MLP	0,8	0,86	86,02	5
Adaboost	FB: age, gender. P: anxious, active.	Tree J48	0,8	0,84	84,94	7
		Tree LMT	0,78	0,79	79,56	11
Tree Random Forest	FB: age, gender, job status.	Tree J48	0,87	0,89	89,24	3
	P: anxious, friend focus, humble.	Tree LMT	0,84	0,86	86,02	4
		MLP	0,82	0,76	76,34	12
Tree J48	FB: age, gender, job status P: disciplined, active, humble	Tree J48	0,89	0,89	89,24	2
		Tree LMT	0,8	0,83	83,87	9

According to the empirical comparisons done based on the recap Table 3 whose values were the output of classification experiments, we find that the most performing classifiers for our classification problem are trees (both Tree LMT and J48). In fact, the classification given by Tree J48 generated important values of accuracy, starting from 77,41 % in the baseline classification until reaching 92,47 % when using feature selection which is a relevant improvement. In these experiments, we used LogitBoost as a Wrapper method and we applied Greedy step wise as a search method. The output of attribute selection was composed of 5 features which are Melancholy, Leisure_oriented, age, gender and job status. The combination of Facebook features and personality features added value to the accuracy of classification. Therefore, we can take an evidence that the association of those features can be considered as suicidal intention predictors and can serve with the Tree J48 classifier as solution for the problem of “suicide intention detection via Facebook” where the Tunisian people are concerned.

5 Conclusion and Future Work

In this paper, our work generally settles in the context of suicide detection via social media. Since the Tunisian revolution of 14 January due to the Bouazizi immolation, suicide rates were continuously increasing: Working on Tunisians was our first contribution. Moreover, statistics shows that Tunisian people tend to be Facebook users. Thus, considering social media site Facebook as reference in suicide detection was our second contribution in this context we developed a Facebook application for data retrieval to collect data about the selected Tunisian profiles. In addition due to the absence of datasets concerning the suicidal Tunisian cases, we lead investigation in order to gather Facebook accounts of suicidal Tunisian and create our own corpus which may be relevant to the futures researches and this is our third contribution.

We proposed combining the personality features with other Facebook features by reference to our logic regarding some fact derived from literature in psychology. Then, by the usage of the supervised classification methods we performed a baseline classification and feature selection on our corpus. The experimental results were satisfying. The usage of decision tree J48 was accurate for suicidal intention prediction starting from 77,41% in the baseline classification until reaching 92,47% after attribute selection. Therefore our idea of adding Facebook features was beneficial for prediction using the classification methods. Our work can be a first step toward a complete application that solve suicide issues along with the whole process from suicide intention detection until prevention: Performing automatic follow ups of the detected suicidal persons until recovery. Also recommending psychologists automatically to the potential suicidal persons.

References

1. WHO Homepage, http://www.who.int/mental_health/prevention/suicide/background/en/. Accessed 01 Jan 2018
2. Burges, Christopher, J.C.: A tutorial on support vector machines for pattern recognition. In: Kluwer Academic Publishers, Boston. Manufactured in The Netherlands (1998)

3. Statista, <https://www.statista.com/statistics/278414/number-of-worldwide-social-network-users/>. Accessed 07 Dec 2017
4. StatCounter Global, <http://gs.statcounter.com/social-media-stats/all/tunisia>. Accessed 07 Jan 2018
5. Bedrosian, R.C.: Bedrosian PhD
6. Bedrosian, R.C., Beck, A.T.: Cognitive aspects of suicidal behavior. In: ISI Journal Citation Reports (1979)
7. Rainer, J.D.: Genetic factors in depression and suicide. In: American journal of psychotherapy (1984)
8. Ross, C.P.: Mobilizing schools for suicide prevention. In: ISI Journal Citation Reports (1980)
9. Harlow, L.L., Newcomb, M.D., Bentler, P.M.: Depression, self-derogation, substance use, and suicide ideation: lack of purpose in life as a mediational factor. In: Journal of clinical psychology (1986)
10. De Choudhury, M., Gamon, M., Counts, S., Horvitz, E.: Predicting depression via social media. In: Proceedings of the Seventh International AAAI Conference on Weblogs and Social Media (2013)
11. Prigerson, H.G., et al.: Influence of traumatic grief on suicidal ideation among young adults. In: Am. J. Psychiatry (1999)
12. Brubaker, J.R., Kivran-Swaine, F., Taber, L., Hayes, G.R.: Grief-stricken in a crowd: the language of bereavement and distress in social media. In: Proceedings of the Sixth International AAAI Conference on Weblogs and Social Media, June 2012
13. Pete, B., Colombo, W., Scourfield, J.: Machine classification and analysis of suicide-related communication on twitter. In: Proceedings of the 26th ACM Conference on Hypertext & Social Media (2015)
14. Colombo, G., Burnap, P., Hodorog, A., Scourfield, J.: Analysing the connectivity and communication of suicidal users on twitter. In: Elsevier Journal (2016)
15. De Choudhury, M., Kiciman, E., Dredze, M., Coppersmith, G., Kumar, M.: Discovering shifts to suicidal ideation from mental health content in social media. In: Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (2016)
16. Yufei, W., Wan, S., Paris, C.: The role of features and context on suicide ideation detection. In: Proceedings of the Australasian Language Technology Association Workshop (2016)
17. Tausczik, Y.R., Pennebaker, J.W.: The psychological meaning of words: LIWC and computerized text analysis methods. *J. Lang. Soc. Psychol.* **29**(1), 24–54 (2010)



Relationships and Sentiment Analysis of Fictional or Real Characters

Paul Diac¹(✉), Cătălina Mărănduc^{1,2}, and Mihaela Colhon³

¹ Faculty of Computer Science, Al. I. Cuza University, Iasi, Romania

paul.diac@info.uaic.ro

² Academic Linguistics Institute I. Jordan – Al. Rosetti, Bucharest, Romania

³ Department of Computer Science, University of Craiova, Craiova, Romania

Abstract. In a previous work, we developed a tool that automatically extrapolated triggers, i.e. diagnostic words for sentiments and relationships, from a manually annotated corpus, the Romanian version of the novel “*Quo Vadis*” by Henryk Sienkiewicz. The NodeXL program can draw graphs of character relationships, to analyse relationships both in the fictional and the real-world. In this research, we describe how we have refined our tool, which becomes both a detector and a semiautomatic (interactive, assisted) annotator of relationships in any previously morphologically annotated real or fictional story. We will also show how we improved and restructured the list of triggers manually annotated in the novel “*Quo Vadis*”. Finally, the tool will annotate the triggers in the Chat corpus, having 2,575 sentences, part of the UAIC Romanian Dependency Treebank, a balanced corpus that contains especially non-standard Romanian language. Finally, we have made graphs to analyse the relations and sentiments of communicators from the Chat corpus.

Keywords: Fictional characters · Graph of relations · Interactive framework · Real characters · Sentiment analysis · Semantic annotation · Social-media communication · Trigger

1 Introduction

Nowadays, text mining applications have to implement deep and meaningful representation of texts which usually implies discovering the entities described in texts and the relations between them. In 2012 our NLP group started a project called “*Quo Vadis*” dedicated to the semantic relations described in texts. It used the Romanian translation of Henryk Sienkiewicz’s novel “*Quo Vadis*”, morphologically annotated previously. The aim of this project was to design a manually annotated corpus, with semantic data, and then, to build an automatic recognizer based on the annotations of the corpus.

The semantic annotations in the Quo Vadis project mark more types of relations mentioned in the text and the entities linked by these relations. The annotations of the relations are defined by the two boundaries of the relations, the type and subtype of the relations, the two arguments (all relations being binary), and the trigger - a word or an expression which signals the relation.

During the Quo Vadis project, the following resources and programs have been created:

- The “*Quo Vadis*” corpus¹, which is publically available on the NLP-Group@UAIC-FII site. It consists of 7,281 sentences manually annotated with semantic data. They consist of marking the textual realization of entities (persons and gods), and also marking four types of semantic relations: *referential*, *affect*, *kinship* and *social* relations. Each type has more subtypes, see [5–7];
- A Web Interface² for visualizing a unique type of annotation: *the co-reference relations between entities*;
- A recognizer for the semantic relations that occur between nested entities [2], i.e. entities that can include one or more other entities. Example: “*the sister of my mother*” is an entity which includes another entity, i.e. “*my mother*”.

Based on the semantic relations described in text, a summary can be automatically generated as illustrated in [4].

In 2014 a trigger detector was built, which memorized the triggers annotated by human annotators. The program generated a list of suggested triggers in certain contexts, in the entire novel. The 5,136 suggestions were validated or invalidated by human annotators. They validated 305 kinship relations, 2,315 social, and 1,219 affect relations, a total of 3,839 that includes also the 757 manually annotated triggers. The percent of validated triggers was 74% from the suggestions [3].

An improved version of this tool will be described in the presented paper. Next, we aim to verify whether the same annotated relationships in the Quo Vadis project (a fictional world) also work in the real world, between chat communicators. The first experiments have been made by selecting only one type of the relationships annotated in the Quo Vadis corpus, namely the detection of affective relationships, which simultaneously leads to a way of sentiment analysis.

We have worked on a Chat corpus, containing 39,391 words and punctuation elements, with the average 15.25 items per sentence. The corpus is morphologically and syntactically annotated, and entirely manually checked. The length of sentences, unusual in chatting, is explained by the high level of education of communicators. A POS-tagger and a syntactic parser were trained on chats and the morphological, lexical, syntactic, semantic, discursive particularities of this type of communication were analysed in [16, 17].

The main contributions of this work can be summarized as follows:

- We propose a more balanced list of semantic relations taking into account the various sentiments and feelings that can be described in various texts;
- We analyse the realization of semantic relations in the non-standardized social media language, a conversation resulting in a less-structured text. The chat style is informal, does not obey any rules;

¹ <http://nlptools.info.uaic.ro/Resources.jsp>.

² <http://nlptools.infoiasi.ro>.

- We propose a tool which brings together the old trigger detector with a framework for the assisted semi-automatic annotation of the relations in the fictional and nonfictional stories;
- Using graphs made with the NodeXL program, as in another work [3], the similarity of the relationships structure in the fictional and real worlds were shown, as in [11]³.

The paper is organized as follows: the first section introduces the actual experiment as a further instalment and sequel of our previous work. The Related Work section summarizes the existing studies conducted in the domain of semantic data. The next section describes the tool that we have designed in order to annotate the chat corpus in a similar way to the annotations in the Quo Vadis corpus. Some statistics given in graphical form are presented in the following section. The article ends with the final conclusions and proposes future research directions.

2 Related Work

Lately, many text analysing applications have implemented semantics in their processing. A continuing growing domain that greatly exploits the semantic data extracted from text is *sentiment analysis* or *opinion mining*; they aim to identify the emotion expressed in texts. The basic goal of sentiment analysis is to identify the overall polarity of a document: positive, negative, or neutral [15].

The semantic oriented approaches usually exploit the relationship between words. State-of-the-Art studies mainly exploit term extraction methods to obtain concepts from texts [18]. In this paper, the semantic relationships between words are identified by a dependency parsing process. Paper [12] explored a new direction in the concept mining field by means of lexicon-syntactic patterns.

Sentiment analysis considers only a special kind of text, namely *affective text*, with the intended aim of analysing the emotional content of texts. The *affective text* analysis has been a popular topic of research in Natural Language Processing (NLP) and Semantic Web communities in recent years [14]. This is an open research problem, relevant for numerous NLP studies such as news stories, public blogs or forums or product reviews [13, 20].

Sentiment analysis tasks are usually designed around an already existing lexicon making use of the WordNet [9], WordNet Affect [19] or ANEW (Affective Norms for English Words) [1]. However, there are still limitations, e.g., WordNet based efforts cannot produce ratings for words not included in WordNet, including multi-word terms and proper nouns [14].

The most important features of the sentiment analysis programs are greatly determined by the quality of the used sentiment lexicons. Other important features included bag-of-word features, hash-tags, handling of negation, word shape and punctuation features, elongated words, etc. [10]. In the lexicon-based approaches, the coverage of the affective lexicon has a great impact on the accuracy scores; consequently, there is a

³ Hansen analyses the social media network and the relationships in Victor Hugo's novel "Les Misérables".

need for methods for automatically updating the lexicon based on the already included elements.

Emotion analysis emerged as a somewhat more specific task than opinion analysis, since it looks at fine-grained types of emotion [10]. Classification of sentences by emotions is done in accordance with some classes of emotions. Here it is worth recalling Ekman's (1992) six classes of emotions: *happiness, anger, sadness, fear, disgust, and surprise*. These emotion classes are the most frequently used ones, being associated with the facial expressions [8].

One of the problems in opinion mining systems is that sarcastic or ironic statements could easily trick these systems [10]. For an automatic system, it is not so important to distinguish between the two of them as it is to eliminate these cases in order to identify the real sentiments that are hidden by these kind of expressions.

3 Bipolar System of Annotation

The interaction with the tool obliged us to adopt a more symmetrical tagset of annotation. We tried not to get away from the annotation of the Quo Vadis corpus (which also underwent changes along the way). But since all relationships are polarized, it is useless to add prepositions such as “of” or “by”. *Rec-love* and *rec-hate* relationships result from the summation of two relationships in which the arguments change their place. We have added the following tags for marking sentiments: LIKE, FEARLESS, OFFEND as the negations of the tags UPSET, FEAR, WORSHIP, because the Quo Vadis system was not symmetrical. The trigger detector has been programmed to suggest us both a sentiment and its negation, in order to select one of them. The problem that remains to be solved is how to formulate rules such that an automatic trigger recognizer can detect irony and sarcasm without being assisted by a human annotator. Example:

In the sentence “*John messed up everything. He is a very intelligent person.*” the trigger detector will suggest the human assistant the annotation of “*intelligent*” as an AFFECT.WORSHIP trigger. The human will choose the button NO and immediately the trigger detector will offer the assistant the opposite trigger: AFFECT.OFFEND. The human will choose the button YES, because it is an ironical statement. The first argument of the trigger is the emitter of the statement (marked at the beginning of the sentence and annotated with the id 0), and the second pole of the relation is John.

If the negation of a feeling does not result in its opposite (a situation in fact), the human annotator will select NO in both alternatives. If a person does not love a particular person, this does not necessarily mean that she/he hates the respective person.

If, by the negation of an AFFECT it results that a person has no AFFECT, we will not annotate anything, because in this project we only deal with the AFFECT annotations.

In fact, only three positive AFFECT relationships and three negative ones are annotated in the Quo Vadis project. By renouncing the targeted variants of the six relationships, we have added 4 other AFFECT relationships, so that the palette of sentiments becomes more comprehensive and the system can be applied to other stories than the one in the novel Quo Vadis, really dominated by the listed feelings (Fig. 1a).

Another preliminary statement is that using this tagset, a limited number of feelings can be annotated. As in reality their range is very wide, each of the ten feelings is

QUOVADIS RELATIONSHIPS TAGSET		OUR RELATIONSHIPS TAGSET	
POSITIVE	NEGATIVE	POSITIVE	NEGATIVE
LOVE	HATE	LOVE	HATE
LOVED BY	HATED BY		
REC-LOVE	REC-HATE		
	FEAR	FEARLESS	FEAR
	FEAR BY		
	UPSET	LIKE	UPSET
FRIEND OF		FRIENDLINESS	ENMITY
WORSHIP		WORSHIP	OFFEND
WORSHIP BY			

a.

adoration LOVE happiness	anger HATE contempt
naughtiness FEARLESS heroism	horror terror FEAR worry
amazement LIKE joy cheerfulness	stress pain UPSET desolation
solidarity FRIENDLINESS care devotion	envy ENMITY malice
admiration WORSHIP enthusiasm	humiliation OFFEND blasphemy

b.

Fig. 1. a. Comparison between Quo VadisAFFECTs and our affect relations tagset. b. The real AFFECTs and the grid of our tags.

understood here in a very broad sense, so that it can include a multitude of feelings. In fact, we have a grid placed over a continuous of real AFFECTs, and when we make the annotation, we choose the tag that comes closest to it. Our tags form a grid, and we hope every real AFFECT can be placed in the perimeter of one of our tags (Fig. 1.b).

Therefore, the aim of the project is to annotate AFFECTs that are considered generic, positive or negative, and to see their proportion and direction in a certain group of real or fictitious characters. We can also make a positive and negative grading as follows:

- Positive: worship > love > like > fearless.
- Negative: offend > hate > upset > fear.

4 Detector and Assisted Annotator

The tool used in 2014 has been transformed in a multi-functional one. It has a directory called “*configurations*” that contains multiple trigger lists with their type and subtype (Social, Kinship, Affect). In another folder, called “*resources*”, we introduce the XML documents that will be annotated. In this way, linguists can use the interface without requiring programmer assistance in order to annotate any narrative text, short or long, having the following characteristics: novel, chat, bible, blogs, comments, etc., provided they have a previous basic annotation i.e., they are segmented in sentences, and each word has an id, lemma, and the morphological analysis; and they adhere to a specific format.

To make the program run, we have to choose a configuration, and then to choose one or more XML files in which the tool will search for all the triggers listed in that configuration. The detector will display the proposals in the order they appear textually, and will ask questions to which the answer is YES or NO, e.g.:

- “Is trigger (**râde**, AFFECT.LIKE) valid for sentence 10_chat2_1500U?” YES / NO

The next step of the assisted annotation, is the proposal of the opposite trigger, if the option NO has been selected. In ironic or sarcastic uses, the opposite is suggested and can be validated:

- “Is trigger (**râde**, AFFECT.HATE) valid for sentence 10_chat2_1500U?” YES / NO

The first variant of the trigger detector was searching in the text form (the list including MWEs, negations, reflexive pronouns, etc. that have been manually annotated in Quo Vadis), which resulted in a high accuracy of detection because some polysemous words are triggers if they have certain neighbourhoods. Therefore, many triggers can escape our detection if they have small formal differences from those found in Quo Vadis. Cases will be numerous, especially if we annotate non-standard texts, such as social media or old Romanian. So, we decided that the new variant of the trigger detector should make searches for trigger **lemmas** resulting in less accuracy, that will lead to a large number of rejected proposals.

By programming the interface for trigger suggestions with both a tag and with its opposite, we eliminated the need of including the verb + negation in the trigger list.

We have created only two types of conditions:

1. if the next word is *xxx*
2. if one of three words above has lemma”sine”.

The first condition led us to detect the words which are triggers only if followed by a certain preposition.

Example:

- The word **ține** is a trigger for AFFECT.LOVE if it is followed by the word **la** as in “**ține la** cineva” (in English, *love somebody*).
- As opposite, “**ține în mână**” (in English, *holds in his hand*) is not a trigger for AFFECT.LOVE.

The second condition helped as detect verbs which are triggers only if preceded by a reflexive pronoun.

If we chose YES option at one of the two first steps, then in the third step the interface displays drop-down lists of the words in the sentence, from which the human annotator chooses: the first and the second poles of the trigger. The relation is a vector from the first to second. Once the human annotator saves his choice, the program will add the annotation in the XML, after the line of the word validated as a trigger, having the following form, similar to those in the Quo Vadis project:

- < relation from = "15" to = "17" trigger = "AFFECT.love"/ >

Were “*from = 15*” indicates the id of the first relation argument or the source, and “*to = 17*” represents the id of the second argument or the target.

The “*trigger = AFFECT.love*” gives the information about the type and subtype of the validated trigger.

By applying similar annotations in multiple resources, we will create a training corpus for the future automatic trigger and argument recognizer.

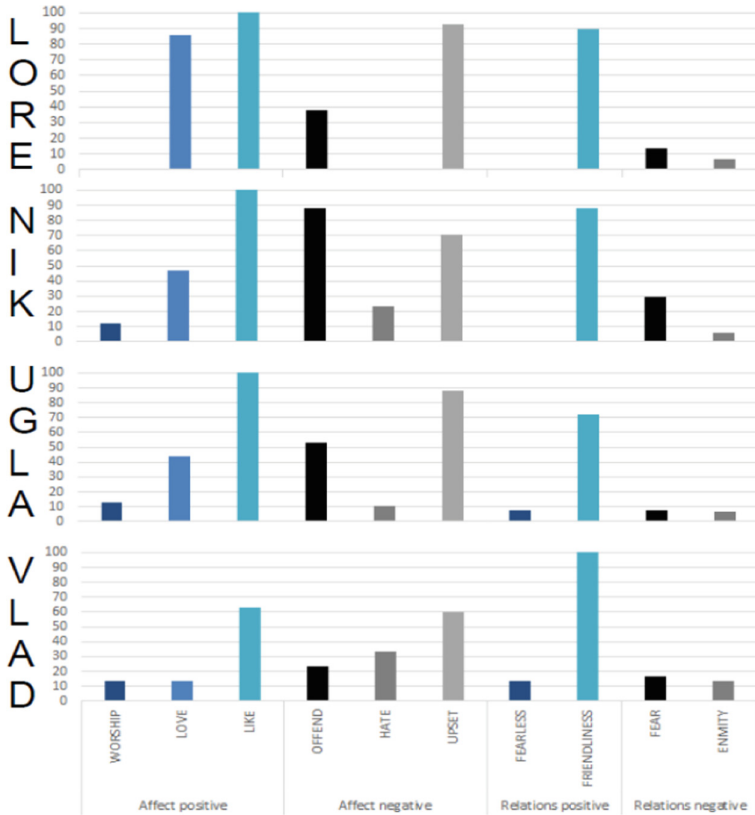


Fig. 2. The frequency of the 10 analyzed affect relations for the 4 communicators.

5 Semantic Relation Graphs

Once we have annotated chats and obtained a consistent training corpus for future triggers and argument recognizer, we have also extracted data from the annotated chat corpus in order to interpret them.

The data extraction consists of the automatic transformation of ids into character names as well as the unification of multiple names for a single entity. The difficulties arise from the fact that the chat corpus is not yet annotated with the names of entities and with their co-references, such as Quo Vadis. The entities that appear as first and second person pronouns must equate with the name of communicators: the phrase issuer with the first person pronouns, and the recipient with the second person pronouns. The communicator's names appear in the id of the sentence, and are annotated with id 0, 1, 2, 3.

We computed the frequency of the 10 feelings for each of the four communicators in our chat corpus: Uglya, 62, researcher, Nik, 61, writer, Lore, 28, psychologist and Vlad, 28, economist.

For Uglya, the predominant feelings are *like* and *upset*, but she also has *fearless* and *offend* relations, she communicates without any reticence, and without bad opinions about anyone. *Fearless* and *offend* also characterize Nik. Lore *hates* nobody, the greatest number of *hate* relations are directed towards Vlad. He has the record number of *friendliness* relations, and of *upset* sentiments, but surprisingly few AFFECT of the subtype *love*, that are dominant to Lore. (Fig. 2).

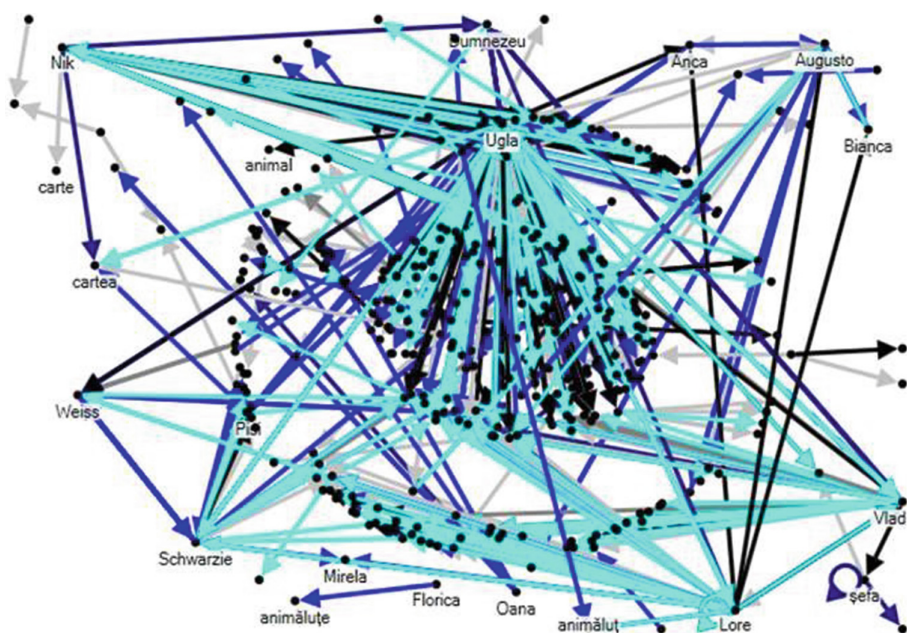


Fig. 3. Graph of AFFECT relations

In order to obtain more suggestive graphs, we decided that *worship*, *love*, *like* (and their negations) are a gradient of positive or negative sentiments, while *friendliness* and *fearless* with their negatives are names of relationships between characters.

The tool annotates the triggered relation in the XML, and saves the data in an XLX table. On the first column, the first selected argument, the source of affect relationship (from...), on the second column, the second argument selected by the user, the target of the relationship (to...): the trigger, their type and subtype are saved on the 3–5 columns and the id of the sentence is saved on the last column. The two poles of a relation can be selected by the user from the words that have the morphological category annotated as noun, pronoun, numeral that does not determine a noun, or possessive adjective.

In this XLX, the trigger lemmas are replaced by the type and subtype, and the argument ids are replaced with the names of communicators or of other persons and objects they refer to. The XLX is introduced in the NodeXL program in order to draw out graphs of character relationships [11].

In the chart of Fig. 3, illustrating more relations, we see a conglomerate of AFFECTs such as *like* and *offend* around Ugly, and a large fascicle of common AFFECTs of subtype *like* with Nik. At the bottom of the chart there are two reciprocal *love* relationships between two couples, Schwarzie-Weiss (black and white cats) and Lore-Vlad. On the other extreme of the chart, Augusto is characterized by *love* relationships in several directions. Oana has a friendly relationship with Ugly and Vlad, a relation that Lore rejects. All characters have AFFECTs *like* or *love* directed to animals. AFFECTs of subtype *hate* and *offend* are directed to hierarchical superiors.

It can be observed that all the characters have multiple *friendship* relationships, but they are particular orientations, and are unified only by Ugly. Most *fear* relationships start at Schwarzie and Vlad. Reflexive relationships, from a character to him or herself appear in both graphs as circles.

The common affective relationships between Ugly and Lore have in the center Schwarzie, the black cat. A strong positive mutual relationship, framed as a bow with arrows at both ends is established between Ugly and Vlad, the two characters who quarrel throughout their dialogue. (See Fig. 3).

If we analyze the characters in the Quo Vadis novel as compared to real world communicators, we see a greater variance and a chaotic orientation of directions in expressed relationships at the latter, which are not controlled by an omniscient author.

The sentiments are various and are not focused; in the chat they can result of a concrete real event. The contradictories sentiments and their ironical expression are frequent. The characters have their own circle of relations and feelings, without connection of the feeling of the other characters.

In Quo Vadis, the target of the affective relationships of the characters can be: gods, Vinicius, Ligia, Petronius, Nero, senate, death, Christ, people, family, Seneca, Venus, Actea, etc.; as resulting from our previous research in [3].

The targets of emotional relationships of real communicators are more varied and there are not only characters, but also objects or abstractions among them: plant, installer, stove, trains, conference, editions, wedding, retirement, managers, sponsors, raven, book, foods, academy, ox, pictures, infarction, math, physics, recreation, etc. See Table 1.

Table 1. The table which generates the graph in the Fig. 3 and the chart in the Fig. 2 (small excerpt).

Argument 1	Argument 2	Trigger	Type	Subtype	Sent. id
cat	doctors	run	AFFECT	FEAR	929U
she(cat)	in darkness	bravery	AFFECT	FEARLESS	932U
me(U)	answer	not receive	AFFECT	UPSET	933U
they(dogs)	meat	seems	AFFECT	LIKE	934U
give(L)	rice with meat	delights	AFFECT	LOVE	934U
me(V)	pictures	not want	AFFECT	HATE	1062V
managers	me(V)	alert	AFFECT	FEARLESS	1062V
U	that	devil	AFFECT	OFFEND	1063U
me(V)	pictures	scared	AFFECT	FEARLESS	1067V
me(U)	well	fell	AFFECT	UPSET	1069U
me(U)	you(V)	not miss	AFFECT	HATE	1069U
they(fellows)	me(V)	condemn	AFFECT	ENMITY	1070V
managers	me(V)	say	AFFECT	FEAR	1072V
managers	pictures	indecent	AFFECT	OFFEND	1073U
me(V)	association	personal	AFFECT	FRIENDLINESS	1076V
me(V)	personal	not allowed	AFFECT	FEAR	1077V

6 Conclusions and Future Work

As an interpretation of our analysis, we observed that the real world is characterized by the variety and divergence of relations. Eventually, we could possibly use this observation to distinguish between a real story and a fictional one. The tool could also be used to extract the orientation of the feelings of real characters towards a particular political line or the preference for the consumption of certain brands.

The tool, applied here on Romanian texts, is language-independent, because in the *resources* and *configurations* folders, documents can be added in any language.

In future, the corpus presented in this paper would be diversified by introducing other communicators and then other types of texts, like social media (twitter messages, product reviews, or political comments). The corpus will also be annotated with entities and coreferences, increasing the precision and recall of suggestions. We also intend to annotate with social and kinship relations, entities, and coreferences, the Mateiu Caragiale's novel "*The Old Courtyard Princess*" published in 1915 that is already morphologically and syntactically annotated entirely supervised. By adding it to Quo Vadis and the Chat corpora, we will form a large training corpus for the trigger and arguments recognizer.

As a first step, the trigger list extracted from our corpora will be very flexible and we the framework will permit us to add triggers. The number of occurrences will be kept in the memory and the triggers which will not have utility will be eliminated. In this way, by retaining only the productive ones, we accuracy of the tool will be increase.

The final plan is to fully automate the learning system, trained on partially supervised annotations of these corpora. This will enable annotating with considerable less effort.

References

1. Bradley, M.M., Lang, P.J.: Affective norms for English words (anew): instruction manual and affective ratings. In: Technical report c-1, University of Florida. The Center for Research in Psychophysiology (1999)
2. Colhon, M., Cristea, D., Gîfu, D.: Discovering semantic relations within nominals. In: Trandabăţ, D., Gîfu, D. (eds.) RUMOUR 2015. CCIS, vol. 588, pp. 85–100. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-32942-0_6
3. Colhon, M., Diac, P., Măranduc, C., Perez, C.A.: Quo vadis research areas – text analysis. In: Proceedings of the 10th International Conference Linguistic Resources and Tools for Processing the Romanian Language. Alexandru Ioan Cuza University Publishing House, pp. 45–56 (2014)
4. Colhon, M., Gîfu, D., Cristea, D.: The Quo Vadis Story Telling. In: Proceedings of the the 11th International Conference Linguistic Resources and Tools for Processing The Romanian Language (ConsILR 2015). Alexandru Ioan Cuza University Publishing House, pp. 93–108 (2015)
5. Cristea, D., Dima, G.E., Postolache, O.D., Mitkov, R.: Handling complex anaphora resolution cases. In: Proceedings of the Discourse Anaphora and Anaphor Resolution Colloquium (DAARC 2002) (2002)
6. Cristea, D., et al.: Quo vadis: a corpus of entities and relations. In: Núria, G., Rapp, R., Bel-Enguix, G. (eds.) Language Production, Cognition, and the Lexicon. Springer International Publishing, pp. 505– 543 (2015)
7. Cristea, D., Ignat, E.: Linking book characters. toward a corpus encoding relations between entities. In: Proceedings of the 7th International Conference on Speech Technology and Human-Computer Dialogue (SpeD 2013), pp. 1–8 (2013)
8. Ekman, P.: An argument for basic emotions. *Cognition and Emotion* 6 (1992)
9. Esuli, A., Sebastiani, F.: Sentiwordnet: a publicly available lexical resource for opinion mining. In: Proceedings of the 5th Conference on Language Resources and Evaluation (LREC 2006), pp. 417–422 (2006)
10. Farzindar, A., Inkpen, D.: Natural Language Processing for Social Media. Morgan & Claypool Publishers (2015)
11. Hansen, D.L., Shneiderman, B., Smith, M.A.: Analyzing Social Media Networks with NodeXL. Morgan Kaufmann (2011)

12. Hearst, M.A.: Automatic acquisition of hyponyms from large text corpora. – Association for Computational Linguistics, vol. 2, pp. 539–545 (1992)
13. Hu, M., Liu, B.: Mining and summarizing customer reviews. In: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Association for Computational Linguistics, pp. 168–77 (2004)
14. Malandrakis, N., Potamianos, A., Iosif, E., Narayanan, S.: Distributional semantic models for affective text analysis. *IEEE Trans. Audio, Speech Lang. Process.* **21**(11), 2379–2392 (2007)
15. Pang, B., Lee, L.: Opinion mining and sentiment analysis. *Found. Trends Inf. Retrieval* (2008)
16. Perez, C.A., Mărănduc, C., Simionescu, R.: Including social media – a very dynamic style – in the corpora for processing Romanian language. In: Linguistic Linked Open Data: 12th EUROLAN 2015 Summer School and RUMOUR 2015 Workshop, Sibiu, Romania, 13–25 July 2015, Revised Selected Papers, pp. 139–153 (2016a)
17. Perez, C.A., Mărănduc, C., Simionescu, R.: Social media – processing Romanian chats and discourse analysis. *Computación y Sistemas* **20**(3), 404–414 (2016b). <https://doi.org/10.13053/CyS-20-3-2453>
18. Poria, S., Agarwal, B., Gelbukh, A., Hussain, A., Howard, N.: Dependency-based semantic parsing for concept level text analysis. *Comput. Linguistics Intell. Text Process. CICLing 2014* (2014)
19. Valitutti, R.: Wordnet-affect: an affective extension of wordnet. In: Proceedings of the 4th International Conference on Language Resources and Evaluation, pp. 1083–1086 (2004)
20. Wiebe, J., Mihalcea, R.: Word sense and subjectivity. In: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics. Association for Computational Linguistics, pp. 1065–1072 (2006)



Sentiment Analysis of Code-Mixed Languages Leveraging Resource Rich Languages

Nurendra Choudhary^(✉), Rajat Singh, Ishita Bindlish, and Manish Shrivastava

Language Technologies Research Centre (LTRC), Kohli Center on Intelligent Systems (KCIS), International Institute of Information Technology, Hyderabad, India
{nurendra.choudhary, rajat.singh}@research.iiit.ac.in,
ishita.bindlish@students.iiit.ac.in, m.shrivastava@iiit.ac.in

Abstract. Code-mixed data is an important challenge of natural language processing because its characteristics completely vary from the traditional structures of standard languages.

In this paper, we propose a novel approach called *Sentiment Analysis of Code-Mixed Text (SACMT)* to classify sentences into their corresponding sentiment - positive, negative or neutral, using contrastive learning. We utilize the shared parameters of siamese networks to map the sentences of code-mixed and standard languages to a common sentiment space. Also, we introduce a basic clustering based preprocessing method to capture variations of code-mixed transliterated words. Our experiments reveal that SACMT outperforms the state-of-the-art approaches in sentiment analysis for code-mixed text by 7.6% in accuracy and 10.1% in F-score.

Keywords: Sentiment analysis · Siamese networks · Code-mixed text

1 Introduction

Multilingual societies with decent amount of internet penetration widely adopted social media platforms. This led to the proliferation in usage of code-mixed text. Sentiment analysis of code-mixed data on social media platforms enables scrutiny of political campaigns, product reviews, advertisements and other social trends.

Code-mixed text adopts the vocabulary and grammar of multiple languages and often forms new structures based on its users. This is challenging for sentiment analysis as traditional semantic analysis approaches do not capture meaning of the sentences. Scarcity of annotated data available for sentiment analysis also limit the advances in the field.

In this paper, we aim to solve the limitations and challenges by utilizing a novel unified framework called “*Sentiment Analysis of Code-Mixed Text (SACMT)*”. SACMT model consists of twin Bi-directional Long Short Term

N. Choudhary and R. Singh—These authors have contributed equally to this work.

Memory Recurrent Neural Networks (BiLSTM RNN) with shared parameters and a contrastive energy function, based on a similarity metric on top. The energy function suits discriminative training for energy-Based models [8].

SACMT learns the shared model parameters and the similarity metric by minimizing the energy function connecting the twin networks. Parameter sharing and the Similarity Metric guarantee that, if the sentiment of sentences on both the individual Bi-LSTM networks are same, then they are nearer to each other in the sentiment space, else they are farther from each other. Hence, the representation of *India match jit gayi* (India won the match) and *Diwali ki shubh kamnaye sabko* (Happy Diwali to everybody) are closer to each other and *India match jit gayi* (India won the match) and *Bhai ki movie flop gayi* (Bhai’s movie was a flop) are distant from each other. The learned similarity metric models the sentiment similarity of sentences into a common sentiment space.

Transliteration of phonetic languages, like Hindi, into roman script creates several variations of the same word. For example, “बहुत” (more) can be transliterated as *bahut*, *bohot* or *bohut*. To solve this challenge, we perform a preprocessing step that aims at clustering multiple word variations together using an empirical similarity metric.

The rest of the paper is organized as follows. Section 2 describes the previous approaches in the field. Section 3 demonstrates the datasets. Section 4 explain the architecture of SACMT. Section 5 defines the baselines. Section 6 and 7 present the experimental set-up and results respectively. Finally, Sect. 8 concludes the paper.

2 Related Work

Distributional semantics [10] approach captures the words’ semantics, but loses out on the information of their sequence in the sentence. Another limitation of the technique is that it considers a word immutable. Hence, it is unable to handle spelling errors, out of vocabulary words properly. [12] assigns polarity scores to individual words. The overall sentiment score of the constituent words assigns the sentence’s polarity. Thus, the semantic relation and words’ sequence is lost and this leads to incorrect classification. N-grams limit this problem but do not eliminate it completely.

Another line of research, [7], utilizes character level LSTMs to learn sub word level information of social media text. This information then classifies the sentences using an annotated corpus. The model presents an effective approach for embedding sentences. However, the limitation in the approach here is the requirement of abundant data.

2.1 Siamese Networks

Siamese networks (shown in Fig. 1) help in the contrastive learning of a similarity metric without an extensive dependence on the features of the input. [3] introduced siamese networks to solve the problem of signature verification. Later, [4]

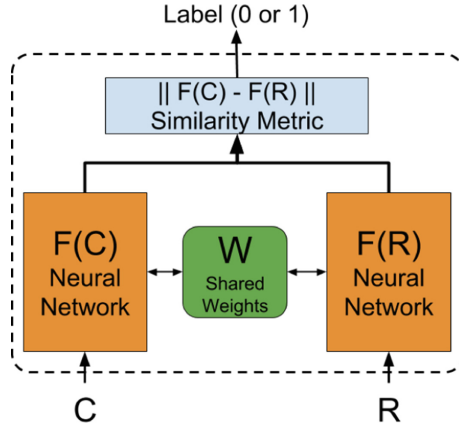


Fig. 1. Siamese Network

used the architecture with discriminative loss function for face verification. These networks also effectively enhance the quality of visual search [6, 9]. Recently, [5] applied these networks to solve the problem of community question answering.

Let, $F(X)$ be the family of functions with parameters W . $F(X)$ is differentiable with respect to W . Siamese network seeks a value of the parameter W such that the symmetric similarity metric is small if X_1 and X_2 belong to the same category, and large if they belong to different categories. The scalar energy function $S(C, R)$ that measures the sentiments' relatedness between tweets of code-mixed (C) text and resource-rich (R) language can be defined as:

$$S(C, R) = \|F(C) - F(R)\| \quad (1)$$

In SACMT, we input the tweets from both the languages to the network. We minimize the loss function such that $S(C, R)$ is small if the C and R carry the same sentiment and large otherwise.

Table 1. Properties of the datasets.

Datasets	Words	Char-trigrams	Positive	Neutral	Negative
HECM	43725	12842	35%	50%	15%
English-Twitter	337913	197649	28%	46%	26%
English-SemEval'13	97280	52011	40%	40%	20%

3 Dataset

We utilize the datasets for testing the architecture on both code-mixed data (Hindi-English) and social media text of a standard language (English). Following are the datasets we considered in our experiments.

- **Hindi-English Code-Mixed (HECM)**: The dataset, proposed in [7], consists of 3879 annotated Hindi-English Code-Mixed sentences.
- **English - Twitter**: The dataset, proposed in [11], consists of 103035 annotated English tweets.
- **SemEval 2013**: The dataset, used for SemEval 2013 Task 2B¹, consists of 11338 annotated English tweets.

All the datasets are annotated with three classes - positive, negative and neutral. Table 1 demonstrates the distribution of classes in the above datasets.

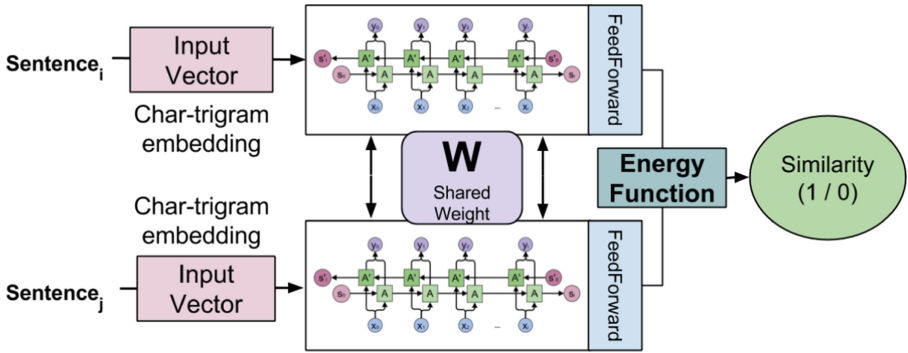


Fig. 2. Architecture of SACMT

4 Architecture of SACMT

As illustrated in Fig. 2, SACMT consists of a siamese network with twin character level Bi-LSTM networks with a fully connected layer on top. Bi-LSTMs project sentences on the two ends to a common sentiment space. We connect the yielded sentiment vectors to a layer that measures the similarity between them. The contrastive loss function combines the similarity measure and the label. Back-propagation through time computes the loss function’s gradient with respect to the weights and biases shared by the sub-networks.

¹ <https://www.cs.york.ac.uk/semEval-2013/task2/index.html>.

Table 2. Some example variations of standard Hindi words with their replacement shown in bold.

Standard	Meaning	Consonants	Captured Variations			
खूबसूरत	beautiful	khbsrt	khoobsurat	khubsurat	khubsoorat	khbsurt
क्यूँकि	because	kynk	kyunki	kiyunki	kiyunkee	kyunkee
मेहरबानी	clemency	mhrbn	meherbani	meharbaani	meharbani	meherbanee
आपका	yours	pk	aapka	apkaa	apka	apkA

4.1 Handling Code-Mixed Word Variations

Transliteration from languages with phonetic script (like Hindi) leads to variation in word depending on the user. We solve this issue using clustering of skip-gram vectors [10]. Skip-gram vectors give the representation of a word in the semantic space based on their context. The variations belong to the same word with similar function implying a similar context. Also, the consonants of these variations in the cases are same (shown in Table 2). Hence, we cluster the words based on a similarity metric that captures both these properties. The similarity metric is formally defined below:

$$f(v1, v2) = \begin{cases} sim(vec(v1), vec(v2)) & \text{if } v1, v2 \text{ have same consonants} \\ 0 & \text{else} \end{cases} \quad (2)$$

where $v1$ and $v2$ are the two variations, sim is a similarity function (like cosine similarity), $vec(v)$ returns the skip-gram vector of v and $f(v1, v2)$ represents the overall similarity between $v1$ and $v2$.

This metric gives us the closest variations for the given word. They together form a cluster and the most frequent word replaces all the other words of the cluster. Here, we assume that the word with the highest frequency also has the most probability of being the correct one.

4.2 Primary Embeddings

Code-mixed text, being informal, has challenges such as spelling errors and out of vocabulary words. These variations cannot be dismissed as errors because they capture sentiment. For example, “*Heeey*” conveys positive sentiment, whereas “*Hey*” conveys a neutral sentiment. Hence, we treat character trigrams as immutable units instead of words. This also reduces the computational complexity as the number of words exceeds character trigrams (shown in Table 1).

We input a pair of character based term vectors of the tweet and a label to the twin networks of SACMT. The label indicates whether the samples are nearer or farther to each other in the sentiment space. For positive samples (nearer in the sentiment space), twin networks are fed with tweets’ vectors with the same sentiment tags. For negative samples (far away in the sentiment space), twin networks are fed with vectors of tweets with different sentiment tags.

4.3 Bidirectional LSTM Network

Each sentence-pair maps into a pair (a_i, a_j) such that $a_i, a_j \in \mathbb{R}^n$ where n is the number of character trigrams in the data.

Bidirectional LSTM [1] model encodes the sequence twice, once forward (original) and once backward (reverse). Back Propagation through Time (BPTT) [2] calculates the weights for both the traversals independently. We apply element-wise Rectified Linear Unit (ReLU) to the output encoding of the BiLSTM. ReLU is defined as: $f(x) = \max(0, x)$. The choice of ReLU simplifies back-propagation, causes faster learning and avoids saturation. The architecture’s final fully connected layer converts the output of the ReLU layer into a fixed length vector $s \in \mathbb{R}^d$. In our architecture, we have empirically set the value of d to 128. The overall model is formalized as:

$$s = \max\{0, W[fw, bw] + b\} \quad (3)$$

where W is a learned parameter matrix (weights), fw is the forward LSTM encoding of the sentence, bw is the backward LSTM encoding of the sentence, and b is a bias term, then passed through an element-wise ReLU.

4.4 Training Step

SACMT differs from the other deep learning counterparts due to its property of parameter sharing, which ensures that both the sentences project into the same sentiment space. Given an input a_i, a_j which are embeddings of tweets and a label $y_i \in \{-1, 1\}$, the loss function is defined as:

$$\text{loss}(a_i, a_j) = \begin{cases} 1 - \cos(a_i, a_j), & \text{if } y = 1; \\ \max(0, \cos(a_i, a_j) - m), & \text{if } y = -1; \end{cases} \quad (4)$$

where m is the margin by which dissimilar pairs should be moved away. It varies between 0 to 1. The loss function is minimized such that pair of tweets with label 1 (same sentiment) are projected nearer to each other and those with label -1 (different sentiment) are projected farther from each other. The model is trained by minimizing the overall loss function in a batch. The objective is to minimize:

$$L(\Lambda) = \sum_{(a_i, a_j) \in C \cup C'} \text{loss}(a_i, a_j) \quad (5)$$

where C contains batch of pairs with same sentiment and C' contains batch of pairs with different sentiment. Back-propagation through time (BPTT) updates the parameters shared by the Bi-LSTM sub-networks.

5 Baselines

Following are the baselines defined according to relevant previous approaches.

- **Average Skip-gram Vectors (ASV):** Word2Vec [10] provides a vector for each word. We average the words’ vectors to get the sentence’s vector. So, each sentence vector is defined as:

$$V_s = \frac{\sum_{w \in W_s} V_w}{|W_s|} \quad (6)$$

where V_s is the vector of the sentence s , W_s is the set of the words in the sentence and V_w is the vector of the word w .

After obtaining each message’s embedding, we train a L2-regularized logistic regression (with ϵ equal to 0.001).

- **Subword LSTM (SWLSTM):** We take the approach, proposed in [7], as the baseline for Hindi-English Code-Mixed data. Character embeddings of the sentence are input and Convolutional Neural Networks capture sub-word level information from the sentence. These embeddings of the tweets classification into different sentiment classes.

6 Experiments

We conduct different experiments to compare the model with diverse inputs and also against the previous approaches in the field. The first experiment (Sect. 6.1) analyzes the performance of SACMT on varying language pairs. In the second experiment (Sect. 6.2), we compare SACMT against the baselines defined in Sect. 5. The third experiment (Sect. 6.3) tests the added performance boost due to the preprocessing step that handles variations. In the final experiment (Sect. 6.4), we provide an extension based on emojis retrieved from social media instead of sentiment tags.

Table 3. Comparison of SACMT trained on different language pairs.

Models	Accuracy	Precision	Recall	F-score
SNASA(HE-HE)	71.3%	0.693	0.668	0.680
SACMT(HE-Eng)	77.3%	0.770	0.749	0.759
SACMT(Eng-Eng)	79.8%	0.795	0.763	0.778

6.1 Experiments for Different Language Pairs

The experiment is a classification task. We consider the Hindi-English Code-Mixed (HECM) sentences and align them with the English sentences from the Twitter datasets of the same sentiment and label them 1 (positive samples). Likewise, we also randomly sample equal number of English sentences with different sentiment (negative samples) and label them -1 . We use this model (SACMT(HE-Eng)) to observe the advantages of training Hindi-English Code-Mixed data in conjunction with English sentences.

Also, we construct the input data by aligning each HECM sentence with corresponding HECM sentences of the same sentiment (positive samples) and label them 1. Likewise, we randomly sample equal number of HECM sentences with different sentiment (negative sample) and label them -1 . Same method constructs the model for English sentences from Twitter dataset. We create these models (SACMT(HE-HE) and SACMT(Eng-Eng)) to observe the advantages that shared parameters of siamese network provide in overall sentiment analysis.

Table 3 demonstrates the performance of these models.

Table 4. Comparison of SACMT with the baselines. ASV and SWLSTM denote the Average Skip-gram vector and Sub-Word LSTM model respectively.

Model	Accuracy	Precision	Recall	F-score
ASV	57.6%	0.5132	0.5336	0.5232
SWLSTM	69.7%	0.646	0.671	0.658
SACMT(HE-HE)	71.3%	0.68	0.665	0.672
SACMT(HE-Eng)	77.3%	0.766	0.753	0.759
Improvement	7.6%	0.12	0.082	0.101

6.2 Comparison with the Baselines

In this experiment, we compare SACMT with the baselines defined in Sect. 5.

We perform contrastive learning of our model using data made by aligning each HECM sentence with a set of English and HECM positive samples (with the same sentiment) with label 1 and a set of negative samples (with different sentiment) of the same size with label -1 . We consider the models SACMT(HE-Eng) and SACMT(HE-HE) for comparison with the baselines.

Both of the above models are evaluated on the HECM dataset. For appropriate comparability, we train and evaluate the baselines on the HECM dataset.

Table 4 demonstrates the performance of baselines and trained models for the experiment.

Table 5. Difference in performance of SACMT with and without the preprocessing step (handling word variations).

Models	With preprocessing				Without preprocessing			
	Accuracy	Precision	Recall	F-score	Accuracy	Precision	Recall	F-score
ASV	59.7%	0.5893	0.5597	0.5741	57.6%	0.5132	0.5336	0.5232
SWLSTM	71.2%	0.669	0.692	0.680	69.7%	0.646	0.671	0.658
SNASA(HE-HE)	72.4%	0.713	0.694	0.703	71.3%	0.693	0.668	0.680
SACMT(HE-Eng)	78.0%	0.775	0.759	0.767	77.3%	0.770	0.749	0.759

6.3 Affect of Handling Word Variations

To analyze the impact of handling word variations on the overall sentiment analysis task. We train all the models defined (including the baselines), both on the regular data and preprocessed data. The difference in the performance is given in Table 5.

Table 6. Distribution after mapping Emojis to respective sentiment classes.

Emojis	Class	Eng	Spa	Hin	Tel
❤️ 😄 😊 😁	Positive	37%	36%	39%	39%
😐 😏 😬 😇	Neutral	31%	30%	31%	31%
😡 😢 😭 😠	Negative	32%	34%	30%	30%

Table 7. Performance enhancement due to emojis in sentiment analysis.

Dataset	SNASA		Emoji-SNASA	
	A(%)	F1	A(%)	F1
HECM	71.3%	0.680	74.8%	0.74
HECM-English	77.3%	0.759	81.5%	0.80
English	79.8%	0.795	82.25%	0.81

6.4 Emoji Based Approach with SACMT (Emoji-SACMT)

In our previous experiment (Sect. 6.1), we observed that in several test scenarios, limited correlation between the language pair leads to incorrectly classified tweets. Emojis are characters used in social media to communicate context inexpressible by normal characters. A major application of these emojis is expressing sentiment. So, we use the emojis available in our social media datasets to align language pairs instead of sentiment tags. Three annotators manually classify the emojis in the dataset into sentiment classes. We only consider the emojis if all the three annotators are in agreement. The distribution of the formed sentiment classes is given in Table 6.

We align each English sentence with a set of positive samples (with the same emoji) with label 1 and a set of negative samples (with different emoji) of the same size with label -1 . The results for the experiment are given in Table 7.

7 Evaluation of the Experiments

From the first experiment’s results (Table 4), we observe that SACMT(Eng-Eng) outperforms the other language pairs. Eng-Eng has the most number of training samples. This presents the significant impact of the training samples’ number on the architecture.

In the second experiment, we observe that SACMT outperforms the state-of-the-art approaches by 7.6% in accuracy and 10.1% in F-score. The additional advantage of shared parameters project the sentences into sentiment space in conjunction with each other. The shared parameters create sentence representations, in accordance to the similarity metric specific to the problem.

Also, we observe that SACMT utilizes language with higher resources to improve the performance of sentiment analysis in the code-mixed text significantly. This allows us to leverage the resources of another language (English in this case) to improve the performance on the code-mixed text.

The third experiment demonstrates the effectiveness of handling word variations. We observe a boost in performance of both the previous approaches and the proposed model by applying a basic preprocessing step.

Multiple times incorrect correlation between the languages in the pair misclassified a sentence. We corrected this behavior by using emojis in twitter dataset to increase the number of usable sentences in establishing correlation. To verify this behavior, we conducted another experiment in Sect. 6.4 to approach this from the perspective of emojis instead of sentiment tags. The experiment's results (given in Table 7) demonstrate that emojis lead to better accuracy. This is seen because emojis lead to a better correlation between the pair's languages. However, the drawback of this approach is that emojis do not always represent perfect sentiment and hence will increase the performance only if the data taken has limited noise.

8 Conclusions

In this paper, we propose SACMT for sentiment analysis of code-mixed text which solves the problem by using shared parameters to project the sentences into a common sentiment space. SACMT employs twin Bidirectional LSTM networks with shared parameters to capture a sentiment based representation of the sentences. We used these sentiment based representations in conjunction with a similarity metric to group sentences with similar sentiment together.

Experiments conducted on the datasets reveal that SACMT outperforms the state-of-the-art approaches significantly. SACMT leverages the resources of other languages to improve the sentiment analysis' performance on code-mixed text.

The word variations' handling, also further, increased performance of all the trained models, including baselines. An emoji based approach used in conjunction with SACMT boosts the performance of overall sentiment analysis further.

As part of future work, we would like to investigate more tasks solvable using resource rich languages as a leverage.

References

1. Barbieri, F., Ballesteros, M., Saggion, H.: Are emojis predictable? arXiv preprint [arXiv:1702.07285](https://arxiv.org/abs/1702.07285) (2017)
2. Boden, M.: A guide to recurrent neural networks and backpropagation. The Dallas project (2002)
3. Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., Shah, R.: Signature verification using a "siamese" time delay neural network. In: Advances in Neural Information Processing Systems, pp. 737–744 (1994)
4. Chopra, S., Hadsell, R., LeCun, Y.: Learning a similarity metric discriminatively, with application to face verification. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005, vol. 1, pp. 539–546. IEEE (2005)

5. Das, A., Yenala, H., Chinnakotla, M., Shrivastava, M.: Together we stand: Siamese networks for similar question retrieval. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). vol. 1, pp. 378–387 (2016)
6. Ding, S., Cong, G., Lin, C.Y., Zhu, X.: Using conditional random fields to extract contexts and answers of questions from online forums. In: ACL. vol. 8, pp. 710–718 (2008)
7. Joshi, A., Prabhu, A., Shrivastava, M., Varma, V.: Towards sub-word level compositions for sentiment analysis of hindi-english code mixed text. In: COLING, pp. 2482–2491 (2016)
8. LeCun, Y., Huang, F.J.: Loss functions for discriminative training of energy-based models. In: AISTats (2005)
9. Liu, Y., Li, S., Cao, Y., Lin, C.Y., Han, D., Yu, Y.: Understanding and summarizing answers in community-based question answering services. In: Proceedings of the 22nd International Conference on Computational Linguistics-vol 1, pp. 497–504. Association for Computational Linguistics (2008)
10. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013)
11. Mozetič, I., Grčar, M., Smailović, J.: Multilingual twitter sentiment classification: the role of human annotators. PloS One **11**(5), e0155036 (2016)
12. Mukku, S.S., Choudhary, N., Mamidi, R.: Enhanced sentiment classification of telugu text using ml techniques. In: SAAIP@ IJCAI, pp. 29–34 (2016)



Emotions Are Universal: Learning Sentiment Based Representations of Resource-Poor Languages Using Siamese Networks

Narendra Choudhary^(✉), Rajat Singh, Ishita Bindlish, and Manish Shrivastava

Language Technologies Research Centre (LTRC), Kohli Center on Intelligent Systems (KCIS), International Institute of Information Technology, Hyderabad, India

{narendra.choudhary, rajat.singh}@research.iiit.ac.in,
ishita.bindlish@students.iiit.ac.in, m.shrivastava@iiit.ac.in

Abstract. Machine learning approaches in sentiment analysis principally rely on the abundance of resources. To limit this dependence, we propose a novel method called *Siamese Network Architecture for Sentiment Analysis (SNASA)* to learn representations of resource-poor languages by jointly training them with resource-rich languages using a siamese network.

SNASA model consists of twin Bi-directional Long Short-Term Memory Recurrent Neural Networks (Bi-LSTM RNN) with shared parameters joined by a contrastive loss function, based on a similarity metric. The model learns the sentence representations of resource-poor and resource-rich language in a common sentiment space by using a similarity metric based on their individual sentiments. The model, hence, projects sentences with similar sentiment closer to each other and the sentences with different sentiment farther from each other. Experiments on large-scale datasets of resource-rich languages - English and Spanish and resource-poor languages - Hindi and Telugu reveal that SNASA outperforms the state-of-the-art sentiment analysis approaches based on distributional semantics, semantic rules, lexicon lists and deep neural network representations without shared parameters.

Keywords: Multilingual sentiment analysis · Contrastive learning

1 Introduction

With proliferation of the Internet into multilingual communities, the linguistic diversity of the real world is being reflected in the virtual world too. Opinionated data like reviews and recommendations are a crucial source of critical analysis for businesses looking for customer experience, expansion into a new segment or their general perception in the market. The data also significantly impacts political policies and campaigns as they represent the public perspective.

N. Choudhary and R. Singh—These authors have contributed equally to this work.

© Springer Nature Switzerland AG 2023

A. Gelbukh (Ed.): CICLing 2018, LNCS 13397, pp. 115–128, 2023.

https://doi.org/10.1007/978-3-031-23804-8_10

Sentiment analysis or polarity detection is a widely studied field in natural language processing with several approaches ranging from rule-based systems to deep learning architectures. Deep learning approaches proved exceptionally effective in solving the task. However, a primary component necessary for the effectiveness of these deep learning approaches is the abundance of data. Hence, major deep learning architectures do not yield satisfactory results in languages with scarce resources. Hence, to overcome the problem, we leverage the abundant resources available in other languages and map both the languages to a common sentiment space.

In this paper, we propose a unified architecture called *Siamese Network Architecture for Sentiment Analysis (SNASA)*. The model consists of twin bi-directional LSTM networks with shared parameters, joined together by a contrastive loss function. The energy function suits the discriminative training for energy based models [14].

SNASA model starts with a simple primary representation based on character trigrams. The model then learns the sentence representation by utilizing the similarity based contrastive energy function. The contrastive function maps the sentences into the sentiment space, such that the distance between sentences with same sentiment is minimized and distance between sentences with different sentiment is maximized. For example, “I am very happy.” and “यह बहुत अच्छी किताब है” (This is a very good book) are closer to each other, whereas, “This is the worst day” and “बगीचा सुन्दर है” (The garden is beautiful) are farther from each other in the sentiment space.

SNASA is a siamese network with shared parameters. We utilize the shared parameters to learn the sentiment based representation for languages with poor resources by jointly training them with resource-rich languages. The model, thus, establishes a correlation between the resource-rich and resource-poor language and maps them to the same sentiment space. This correlation is further utilized to predict the sentiment of the resource-poor languages using the immense data available in resource-rich languages.

The rest of the paper is organized as follows. Section 2 presents the previous approaches to conquer the problem. Section 3 describes the evaluation dataset and Sect. 4 describes the architecture of SNASA. Section 5 explains the training and testing phase of SNASA. Section 6 details the baselines. Section 7 presents the experimental set-up and results. Finally, Sect. 8 concludes the paper.

2 Related Work

Sentiment analysis is a widely studied task with various approaches proposed in the recent period. In this section, we survey the previous methodologies for the task.

Distributional semantics [15] approach captures the sentence’s overall semantic value but does not maintain information of the words’ order. [18] propose

classifier models based on support vector machines that assigns sentiment polarity to words or phrases using classifiers. Polarity of its constituents totals the sentences' polarity. Lexicon based approaches [23] utilize a manually constructed lexicon with sentiments of major words given. This information assigns the polarity. The limitation of these approaches is the information loss of the words' sequence which leads to the wrong classification. e.g.; In *"I am not happy"*, *"not"* carries a negative sentiment and *"happy"* carries a positive sentiment. The combination gives a neutral sentiment, whereas the sentence is truly negative. Bag of n-grams limit this effect but do not eliminate it completely.

Matrix Vector Recursive Neural Network (MV-RNN) [22] provides a solution to the problem of capturing the words' relation in a sentence. The model assigns a vector and a matrix to each node of the sentence's syntactically parsed tree. The vector and matrix represent the word's semantic value and its relation with the other words respectively. This approach presents an effective model for capturing the content and relations of the sentence. However, the approach requires a large amount of data to train and hence will fail in case of languages with fewer resources.

Adaboost based Convolution Neural Networks (Ada-CNN) [10] uses CNN classifiers with different filter sizes. Adaboost arrives at a weighted combination of the classifiers. The differing filter sizes analyze the contribution of different n-grams to the overall sentiment.

Another line of research [2, 11] utilizes rules and vocabulary of the languages to classify sentences. These techniques are highly accurate but susceptible to the problems of spelling errors and improper sentences. And these problems are frequent in any informal text including reviews and tweets. Also, in case of Hindi, [21] have trained a multinomial naive bayes model on annotated Hindi tweets to solve the problem.

Additionally, there have been efforts by researchers [19] to generate annotated resources by utilizing available raw corpus. They employ the availability of different domains to construct a Multi-arm Active Transfer Learning (MATL) algorithm to label raw samples and continuously add them to the original dataset. Each step updates the algorithm's parameters using reinforcement learning with a reward function. The above approach works well for the considered domains - sports, movies and politics. These domains have a formal vocabulary and grammar, whereas, tweets do not follow this trend. Hence, the model is inapplicable to unstructured tweets. The new resources depend on the available resources' domain, which is risky, especially in the case of tweets that do not comply with any certain domain.

Usually, methods that require the immutable words are ineffective. A better approach utilizes the languages' characters instead of words. Given their proven effectiveness in [1, 6, 8, 9, 13, 24, 25], we use Bidirectional LSTMs (Bi-LSTMs) based on character n-grams. This approach produces embeddings based on the sequence of character n-grams, thus eliminating the problems of spelling mistakes and agglutination (in the case of some languages such as Telugu).

Although Bi-LSTMs map the sentences to a sentiment space, we also require the distance between the sentences with the similar sentiment to be closer and the sentences with the different sentiment to be farther. For this reason, we use the architecture of Siamese Networks. This architecture possesses the capability of learning similarity from the given data without requiring specific information about the classes.

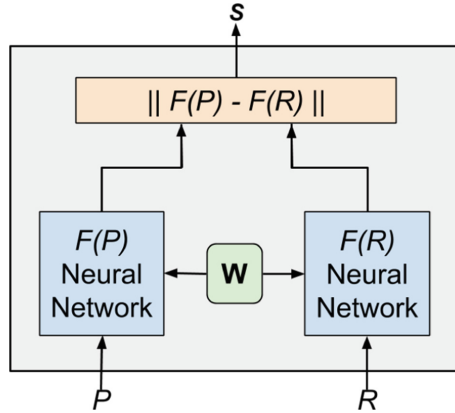


Fig. 1. Siamese Network

2.1 Siamese Networks

[4] introduced siamese neural networks to solve the problem of signature verification. Later, [5] used the architecture with discriminative loss function for face verification. Recently, these networks solved the problem of community question answering [7]. Let, $F(X)$ be the family of functions with parameters W . $F(X)$ is differentiable with respect to W . Siamese network seeks a value of the parameter W such that the symmetric similarity metric is small if X_1 and X_2 belong to the same category, and large if they belong to different categories. The scalar energy function $S(R, P)$ that measures the relatedness of sentiments between resource-poor (P) and resource-rich (R) language's tweets can be defined as:

$$S(P, R) = \|F(P) - F(R)\| \quad (1)$$

In SNASA, we input the tweets from both the languages to the network. The loss function is minimized so that $S(P, R)$ is small if the R and P carry the same sentiment and large otherwise.

3 Datasets

The datasets for different languages are given below:

Table 1. Distribution of the datasets considered in the experiments. Pos, Neg, Neu, V. Pos and V.Neg stand for Positive, Negative, Neutral, Very Positive and Very Negative respectively. 4 classes are available only in Movie Review dataset.

Datasets	Sentence length	3 classes			4 classes			
		Pos	Neg	Neu	V.Pos	Pos	Neg	V.Neg
English - Movie Reviews	429	38%	24%	38%	17%	40%	31%	12%
English - Twitter	12	29%	26%	45%	–	–	–	–
Spanish - Twitter	14	48%	13%	39%	–	–	–	–
Hindi - Reviews	15	33%	31%	36%	–	–	–	–
Telugu - News	13	27%	27%	46%	–	–	–	–

- **English - Movie Review Dataset:** The dataset [20] consists of 5006 movie reviews annotated into 3 classes (positive, neutral and negative) and 4 classes (very positive, positive, negative and very negative).
- **English - Twitter Dataset:** The dataset [17] consists of 103035 tweets annotated into 3 classes - positive, neutral and negative.
- **Spanish - Twitter Dataset:** The dataset [17] consists of 275589 tweets annotated into 3 classes - positive, neutral and negative.
- **Hindi - Product Review Dataset:** The dataset [16] consists of 1004 product reviews annotated into 3 classes - positive, neutral and negative.
- **Telugu - News Dataset:** The dataset [19] is an annotated corpus of news data tagged into 3 classes - positive, neutral and negative.

The sentiment tags' distribution in the above datasets is given in Table 1.

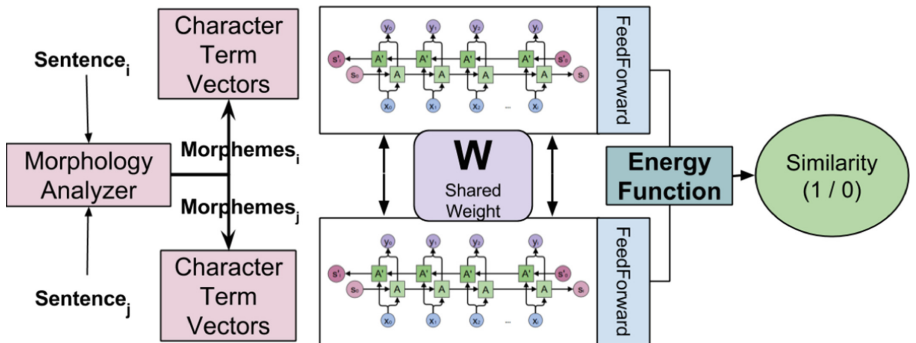


Fig. 2. Architecture of SNASA

4 Architecture of SNASA

As shown in Fig. 2, SNASA consists of a Bi-LSTMs pair and a dense feed forward layer at the top. The Bi-LSTMs capture the sequence and constituents of the sentence and project them to a sentiment space. We connect the yielded sentiment vectors to a layer that measures similarity between them. The contrastive loss function combines the similarity measure and the label. Back-propagation through time computes the loss function’s gradient with respect to the weights and biases shared by the sub-networks.

Table 2. Number of Unique Character Trigrams and Words in the datasets

Language	Tel-News	Hin-Reviews	Spa-Twitter	Eng-Twitter	Eng-Movie Review
Char trigrams	17424	6059	296797	197639	13897
Words	75417	10244	481280	359113	2148164

4.1 Primary Representation

Informal data consists of a lot of spelling errors, out-of-vocabulary(OOV) words and multiple spelling of the same word. The way of writing a word may also convey a sentiment (e.g.; “Hiiii” conveys a positive sentiment whereas “Hi” is a neutral sentiment). Hence, we use character trigrams to embed the sentence instead of using words. This approach takes care of the spelling errors and OOV words because a partial match exists in the character trigrams. Character trigrams take the information of all the inflections of a word, thus, eliminating the problem of agglutination. This method, also, captures the sentiment of different ways of writing as information is attained on a character-level. To further address the problem of agglutination in morphologically rich languages, we add a morphology analyzer that divides the words into its constituent morphemes. This also helps in the computational complexity as the number of character trigrams is far less than the number of complete words (shown in Table 2). The approach represents a sentence using a vector with number of dimensions equal to the number of unique character trigrams in the training dataset.

We input character based term vectors of resource-poor and resource-rich language’s tweets and a label to the twin networks of SNASA. The label indicates whether the samples are nearer or farther to each other in the sentiment space. For positive samples (nearer in the sentiment space), we feed the twin networks with term vectors of tweets (one from resource-poor and one from resource-rich) with the same sentiment tag. For negative samples (far away from each other in the sentiment space), we feed the twin networks with term vectors of tweets (one from resource-poor and one from resource-rich) with different sentiment tags.

4.2 Bi-directional LSTM Network

We map each sentence-pair into $[p_i, r_i]$ such that $p_i \in \mathbb{R}^m$ and $r_i \in \mathbb{R}^n$, where m and n are the total number of character trigrams in the resource-poor language and the resource-rich language respectively.

Bi-LSTM model encodes the sentence twice, one in the original order (forward) of the sentence and one in the reverse order (backward). Back-propagation through time [3] calculates the weights for both the orders independently. The algorithm works in the same way as general back-propagation, except in this case the back-propagation occurs over all the hidden states of the unfolded timesteps.

We, then, apply element-wise Rectified Linear Unit (ReLU) to the output encoding of the BiLSTM. ReLU is defined as: $f(x) = \max(0, x)$. The choice of ReLU simplifies back-propagation, causes faster learning and avoids saturation.

The architecture’s final dense feed forward layer converts the output of the ReLU layer into a fixed length vector $s \in \mathbb{R}^d$. In our architecture, we empirically set the value of d to 128. The overall model is formalized as:

$$s = \max\{0, W[fw, bw] + b\} \quad (2)$$

where W is a learned parameter matrix (weights), fw is the forward LSTM encoding of the sentence, bw is the backward LSTM encoding of the sentence, and b is a bias term, then passed through an element-wise ReLU.

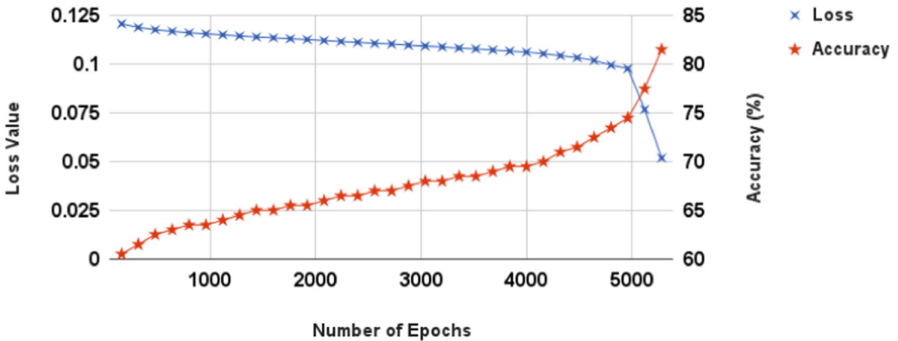


Fig. 3. Number of Epochs vs Loss and Accuracy

5 Training and Testing

We train SNASA on the pairs of sentences in resource-poor and resource-rich language to capture their similarity in the sentiment. SNASA differs from other deep learning counterparts due to its property of parameter sharing. Training the network with a shared set of parameters not only reduces the number of

parameters (thus, save many computations) but also ensures that the sentences of both the languages project into the same sentiment space. We learn the network’s shared parameters to minimize the distance between the sentences with the same sentiment and maximize the distance between the tweets with different sentiment.

Table 3. Comparison between different language pairs for 3 classes and 4 classes (only Movie Review).

Language pair	3 classes				4 classes			
	Accuracy	Precision	Recall	F-score	Accuracy	Precision	Recall	F-score
Eng-Eng	81.25%	0.83	0.80	0.81	66.1%	0.67	0.64	0.65
Eng-Hin	80.5%	0.82	0.79	0.80	–	–	–	–
Eng-Tel	80.3%	0.82	0.79	0.80	–	–	–	–
Eng-Spa	81.5%	0.83	0.80	0.81	–	–	–	–
Hin-Tel	70.2%	0.72	0.69	0.70	–	–	–	–

Given an input p_i, r_i where p_i and r_i are tweets from resource-poor and resource-rich languages respectively and a label $y_i \in \{-1, 1\}$, the loss function is defined as:

$$l(p_i, r_i) = \begin{cases} 1 - \cos(p_i, r_i), & y = 1; \\ \max(0, \cos(p_i, r_i) - m), & y = -1; \end{cases} \quad (3)$$

where m is the margin that decides the distance by which dissimilar pairs should be moved away from each other. It generally varies between 0 to 1. The loss function is minimized such that pair of tweets with the label 1 (same emoji) are projected nearer to each other and pair of tweets with the label -1 (different emoji) are projected farther from each other in the sentiment space. The model is trained by minimizing the overall loss function in a batch. The objective is to minimize:

$$L(\Lambda) = \sum_{(p_i, r_i) \in C \cup C'} l(p_i, r_i) \quad (4)$$

where C contains the batch of same sentiment sentence pairs and C' contains the batch of different sentiment sentence pairs. Back-propagation through time (BPTT) updates the parameters shared by the Bi-LSTM sub-networks.

For testing, we randomly sample a certain number (100 in our case) of sentences for each sentiment $R_{sentiment}$ from the language corpus with higher amount of data. For every input, we then apply the trained model to get the similarity between the input and all corresponding $R_{sentiment}$. The $R_{sentiment}$ with the most matches with the input is finally selected as the correct polarity tag.

In case the correlated data available for both the resource-rich and resource-poor languages are not annotated, we use the one language’s abundant resources to construct a state-of-the-art sentiment analysis model [10]. The sentiment analysis model in conjunction with the correlation data obtained from SNASA aids the resource-poor language’s prediction.

6 Baselines

The approaches vary based on the language in consideration. Hence, baselines are also defined below accordingly. English, Japanese and Spanish enjoy the highest share of data on Twitter¹. We consider English and Spanish because of their script and typological similarity (both are SVO). The baselines considered for resource-rich languages - English and Spanish are:

- **Average Skip-Gram Vectors (ASV):** We train a Word2Vec skip-gram model [15] on a corpus of 65 million raw sentences in English and 20 million raw sentences in Spanish. Word2Vec provides a vector for each word. We average the words’ vectors to get the sentence’s vector. So, each sentence vector is defined as:

$$V_s = \frac{\sum_{w \in W_s} V_w}{|W_s|} \quad (5)$$

where V_s is the sentence’s vector s , W_s is the set of the words in the sentence and V_w is the vector of the word w .

After obtaining each message’s embedding, we train an L2-regularized logistic regression, (with ϵ equal to 0.001).

- **Matrix Vector Recursive Neural Network (MV-RNN):** The model [22] assigns a vector and a matrix to every node of a syntactic parsed tree. The vector represents the node’s semantic value and the matrix represents its relation with the neighboring words. A recursive neural network model is then trained using backpropagation through structure to define the nodes’ weighted contribution to the sentence’s sentiment.
- **Adaboost Based Convolutional Neural Network (Ada-CNN):** CNN sentence classifier models [12] with filter sizes 3,4 and 5 are trained on the datasets. These filter sizes capture the 3-gram, 4-gram and 5-gram contribution to the overall sentiment respectively. Adaboost then attains a weighted combination of these classifiers. This weighted combination of the classifiers assigns the overall sentiment tag. This helps in giving a weighted emphasis to the information provided by 3-grams, 4-grams and 5-grams in the sentence.

Hindi and Telugu are the 3rd and 17th most spoken language in the world respectively. But they hold a relatively low share of Twitter data. The speakers of Hindi and Telugu on Twitter primarily use the roman transliterated form of the language. This also further translates to a limited availability of annotated corpus for these languages. The baselines for these languages are:

- **Domain Specific Classifier (Telugu) (DSC-T):** We train a Word2Vec model on a corpus of 700,000 raw Telugu sentences provided by Indian Languages Corpora Initiative (ILCI). We train a Random Forest (RF) and Support Vector Machines classifier (SVM) (given by [18]) on the Telugu News dataset to construct our baseline for Telugu language.

¹ The Many Tongues of Twitter - MIT Technology Review.

Table 4. Comparison with the English baselines on Movie Review dataset. ASV is Average Skip-gram Vectors, MV-RNN refer to Matrix Vector Recursive Neural Network model.

	3 classes				4 classes			
Method	Accuracy	Precision	Recall	F-score	Accuracy	Precision	Recall	F-score
ASV	52.59%	0.49	0.52	0.50	39.42%	0.47	0.45	0.32
MV-RNN	79.0%	0.77	0.75	0.76	64.3%	0.63	0.62	0.62
SNASA	81.25%	0.83	0.80	0.81	66.1%	0.67	0.64	0.65

Table 5. Comparison with the baselines on three-class datasets of the respective languages. ASV is Average Skip-gram Vectors, MV-RNN refer to Matrix Vector Recursive Neural Network model. They are baselines for English and compare to SNASA (Eng-Eng). DSC-T is Domain Specific Classifier for Telugu and compares to SNASA (Tel-Eng). MNB-H refers to Multinomial Bayes Model for Hindi and compares to SNASA (Hin-Eng).

Method	Accuracy	Precision	Recall	F-score
ASV	52.59%	0.49	0.52	0.50
MV-RNN	79.0%	0.77	0.75	0.76
DSC-T	68.17%	0.67	0.66	0.66
MNB-H	62.14%	0.61	0.58	0.59
SNASA (Eng-Eng)	81.25%	0.83	0.80	0.81
SNASA (Hin-Eng)	80.5%	0.82	0.79	0.80
SNASA (Tel-Eng)	80.3%	0.82	0.79	0.80

- **Multinomial Naive Bayes Model (Hindi) (MNB-H):** We train a multinomial naive bayes model (given by [21]) on the Hindi Review dataset to form our baseline for Hindi language.

7 Experiments and Evaluation

In order to study the comparison of SNASA to the previous models, we performed an array of experiments. In the first experiment (Sect. 7.1), we analyze varying language pairs and make a comparison between them. In the second experiment (Sect. 7.2), we compare our model against previous approaches in the problem of Sentiment Analysis. In the third experiment (Sect. 7.3), we provide an extension where emojis retrieved from Twitter are utilized instead of regular sentiment tags.

7.1 Experiments for Different Language Pairs

The experiment is a classification task. We take the English and Hindi three-class datasets (Eng-Hin) and align each Hindi sentence with English sentences

of the same sentiment (positive samples) and label them 1. Similarly, we also randomly sample the same number of English tweets with different sentiment (negative samples) for each Hindi Tweet and label them -1.

Similarly, we repeat the experiment for English-Telugu (Eng-Tel) dataset pair, English-Spanish (Eng-Spa) dataset pair, English-English (Eng-Eng) dataset pair and Hindi-Telugu (Hin-Tel) dataset pair. Table 3 demonstrates the results of the experiments.

We run another experiment for the case of English (Eng-Eng), where we take the case of Movie Review dataset and align each sentence with other sentences of the same sentiment (positive samples) and label them 1. Similarly, we also randomly sample the same number of sentences with different sentiment (negative samples) and label them -1. We perform the experiment for both three-class and four-class classification task. The results of this experiment are given in Table 3.

7.2 Comparison with the Baselines

In this experiment, we compare our model against the baselines (defined in Sect. 6).

We defined the baselines for resource-rich languages on English. So, we perform contrastive learning of our model using data made by aligning each English sentence with a set of positive samples (with the same sentiment) with label 1 and a set of negative samples (with different sentiment) of the same size with label -1.

In the case of resource-poor languages, i.e. Hindi and Telugu, we perform contrastive learning of our model using data made by aligning each of the resource-poor language (Hindi and Telugu) sentence with a set of positive English samples (with the same sentiment) with label 1 and a set of negative English samples (with different sentiment) of the same size with label -1.

The baselines on English are trained and evaluated on both Movie Review dataset and three-class dataset. The baselines on Spanish, Hindi and Telugu are trained and evaluated on their respective three-class datasets.

The results of the comparison between SNASA and previous approaches on Movie Review dataset are given in Table 4. The comparison between SNASA and previous approaches on three-class datasets are given in Table 5.

Table 6. Distribution after mapping Emojis to respective sentiment classes.

Emojis	Class	Eng	Spa	Hin	Tel
	Positive	37%	36%	39%	39%
	Neutral	31%	30%	31%	31%
	Negative	32%	34%	30%	30%

Table 7. Performance enhancement due to emojis in sentiment analysis.

	SNASA		Emoji-SNASA	
Dataset	A(%)	F1	A(%)	F1
English	81.25%	0.81	84.8%	0.83
Spanish	81.5%	0.81	85.2%	0.83

7.3 Emoji Based Approach with SNASA (Emoji-SNASA)

In our previous experiment (Sect. 7.1), we found that in several test scenarios, the tweet is incorrectly classified because of limited correlation data available between the language pair. Emojis are characters used in social media to communicate context inexpressible by normal characters. A major application of these emojis is in expressing sentiment. So, we use the emojis available in our datasets to align language pairs instead of sentiment tags. The emojis in the dataset are classified manually into sentiment classes by three annotators. The emojis were taken into consideration only if all the three annotators were in agreement. The distribution of each of thus formed sentiment classes is given in Table 6.

We align each English sentence with a set of positive samples (with the same emoji) with label 1 and a set of negative samples (with different emoji) of the same size with label -1. The results for the experiment are given in Table 7.

7.4 Evaluation of the Experiments

We observe from Table 3 that the best overall results for sentiment analysis are seen for the English-Spanish pair. This is due to the English-Spanish containing the maximum number of tweet pairs. We also note from Fig. 3 that with increasing number of epochs, the accuracy and overall performance considerably gets better.

Multiple times a sentence is misclassified because of incorrect correlation between the languages in the pair. We corrected this behavior using emojis in three-class datasets to increase the number of sentences that could be used to establish correlation. To verify this behavior, we conducted another experiment in Sect. 7.3 to approach this from the perspective of emojis instead of sentiment tags. The experiment’s result (given in Table 7) demonstrate that emojis lead to better accuracy. This is seen because emojis lead to a better correlation between the languages’ pair. However, emojis do not always represent perfect sentiment and hence will increase the performance only if the data taken has limited noise.

From Tables 4 and 5, we observe that SNASA outperforms the current approaches significantly, especially in the case of resource-poor languages. Interestingly, the results also show that using shared parameters leads to an improvement in performance. SNASA learns representation, specifically, for the task of sentiment classification. It leverages the relatively resource-rich language for the improvement in the resource-scarce language’s performance.

8 Conclusions

In this paper, we proposed SNASA for sentiment analysis of resource-poor languages which solves the problem by projecting the resource-poor language and resource-rich language in the same sentiment space. SNASA employs twin Bidirectional LSTM networks with shared parameters to capture a sentiment based

representation of the sentences. These sentiment based representations are used in conjunction with a similarity metric to group sentences with similar sentiment together.

An emoji based approach used in conjunction with SNASA boosts the performance of overall sentiment analysis further. Experiments conducted on three-class and four-class (Movie Review) datasets revealed that SNASA outperforms the current state-of-the-art approaches significantly.

In future, we would like to apply the current model on more applications based on learning similarity like question-answering, conversation systems and semantic similarity. Though, of course, the presence and impact of correlation between languages would be limited in other areas. Also, we believe that there is a good case for integration of attention-based models in the subnetworks.

References

1. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint [arXiv:1409.0473](https://arxiv.org/abs/1409.0473) (2014)
2. Balamurali, A., Joshi, A., Bhattacharyya, P.: Cross-lingual sentiment analysis for indian languages using linked wordnets. In: Proceedings of COLING 2012: Posters, pp. 73–82 (2012)
3. Boden, M.: A guide to recurrent neural networks and backpropagation. the Dallas project (2002)
4. Bromley, J., Guyon, I., LeCun, Y., Säcker, E., Shah, R.: Signature verification using a” siamese” time delay neural network. In: Advances in Neural Information Processing Systems, pp. 737–744 (1994)
5. Chopra, S., Hadsell, R., LeCun, Y.: Learning a similarity metric discriminatively, with application to face verification. In: Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on. vol. 1, pp. 539–546. IEEE (2005)
6. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint [arXiv:1412.3555](https://arxiv.org/abs/1412.3555) (2014)
7. Das, A., Yenala, H., Chinnakotla, M., Shrivastava, M.: Together we stand: Siamese networks for similar question retrieval. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). vol. 1, pp. 378–387 (2016)
8. Dhingra, B., Zhou, Z., Fitzpatrick, D., Muehl, M., Cohen, W.W.: Tweet2vec: Character-based distributed representations for social media. arXiv preprint [arXiv:1605.03481](https://arxiv.org/abs/1605.03481) (2016)
9. Dyer, C., Ballesteros, M., Ling, W., Matthews, A., Smith, N.A.: Transition-based dependency parsing with stack long short-term memory. arXiv preprint [arXiv:1505.08075](https://arxiv.org/abs/1505.08075) (2015)
10. Gao, Y., Rong, W., Shen, Y., Xiong, Z.: Convolutional neural network based sentiment analysis using adaboost combination. In: Neural Networks (IJCNN), 2016 International Joint Conference on, pp. 1333–1338. IEEE (2016)
11. Joshi, A., Balamurali, A., Bhattacharyya, P.: A fall-back strategy for sentiment analysis in hindi: a case study. In: Proceedings of the 8th ICON (2010)
12. Kim, Y.: Convolutional neural networks for sentence classification. arXiv preprint [arXiv:1408.5882](https://arxiv.org/abs/1408.5882) (2014)

13. Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., Dyer, C.: Neural architectures for named entity recognition. arXiv preprint [arXiv:1603.01360](https://arxiv.org/abs/1603.01360) (2016)
14. LeCun, Y., Huang, F.J.: Loss functions for discriminative training of energy-based models. In: AISTats (2005)
15. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems, pp. 3111–3119 (2013)
16. Mogadala, A., Varma, V.: Retrieval approach to extract opinions about people from resource scarce language news articles. In: Proceedings of the First International Workshop on Issues of Sentiment Discovery and Opinion Mining, p. 4. ACM (2012)
17. Mozetič, I., Grčar, M., Smailović, J.: Multilingual twitter sentiment classification: the role of human annotators. PloS one **11**(5), e0155036 (2016)
18. Mukku, S.S., Choudhary, N., Mamidi, R.: Enhanced sentiment classification of telugu text using ml techniques. In: SAAIP@ IJCAI, pp. 29–34 (2016)
19. Mukku, S.S., Oota, S.R., Mamidi, R.: Tag me a label with multi-arm: active learning for Telugu sentiment analysis. In: Bellatreche, L., Chakravarthy, S. (eds.) DaWaK 2017. LNCS, vol. 10440, pp. 355–367. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-64283-3_26
20. Pang, B., Lee, L.: Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In: Proceedings of the 43rd annual meeting on association for computational linguistics, pp. 115–124. Association for Computational Linguistics (2005)
21. Sarkar, K., Chakraborty, S.: A sentiment analysis system for Indian language tweets. In: Prasath, R., Vuppala, A.K., Kathirvalavakumar, T. (eds.) MIKE 2015. LNCS (LNAI), vol. 9468, pp. 694–702. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-26832-3_66
22. Socher, R., Huval, B., Manning, C.D., Ng, A.Y.: Semantic compositionality through recursive matrix-vector spaces. In: Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning, pp. 1201–1211. Association for Computational Linguistics (2012)
23. Taboada, M., Brooke, J., Tofiloski, M., Voll, K., Stede, M.: Lexicon-based methods for sentiment analysis. *Comput. Linguist.* **37**(2), 267–307 (2011)
24. Vinyals, O., Kaiser, L., Koo, T., Petrov, S., Sutskever, I., Hinton, G.: Grammar as a foreign language. In: Advances in Neural Information Processing Systems, pp. 2773–2781 (2015)
25. Wang, P., Qian, Y., Soong, F.K., He, L., Zhao, H.: Learning distributed word representations for bidirectional lstm recurrent neural network. In: HLT-NAACL, pp. 527–533 (2016)



Contrastive Learning of Emoji-Based Representations for Resource-Poor Languages

Nurendra Choudhary^(✉), Rajat Singh, Ishita Bindlish, and Manish Shrivastava

Language Technologies Research Centre (LTRC), Kohli Center on Intelligent Systems (KCIS), International Institute of Information Technology, Hyderabad, India
{nurendra.choudhary, rajat.singh}@research.iiit.ac.in,
ishita.bindlish@students.iiit.ac.in, m.shrivastava@iiit.ac.in

Abstract. The introduction of emojis (or emoticons) in social media platforms has given the users an increased potential for expression. We propose a novel method called *Classification of Emojis using Siamese Network Architecture (CESNA)* to learn emoji-based representations of resource-poor languages by jointly training them with resource-rich languages using a siamese network.

CESNA model consists of twin Bi-directional Long Short-Term Memory Recurrent Neural Networks (Bi-LSTM RNN) with shared parameters joined by a contrastive loss function based on a similarity metric. The model learns the representations of resource-poor and resource-rich language in a common emoji space by using a similarity metric based on the emojis present in sentences from both languages. The model, hence, projects sentences with similar emojis closer to each other and the sentences with different emojis farther from one another. Experiments on large-scale Twitter datasets of resource-rich languages - English and Spanish and resource-poor languages - Hindi and Telugu reveal that CESNA outperforms the state-of-the-art emoji prediction approaches based on distributional semantics, semantic rules, lexicon lists and deep neural network representations without shared parameters.

Keywords: Multilingual emoji prediction · Contrastive learning

1 Introduction

Social media continues to grow exponentially since its inception and has now become a forum filled with people's expression, opinions and sentiments. To better capture the text's sentimental context, users adopted *emojis*. Basically, emojis are special characters (or pictures) used to communicate context inexpressible by standard text. Emojis are ideograms and smileys used in electronic

N. Choudhary and R. Singh—These authors have contributed equally to this work.

© Springer Nature Switzerland AG 2023

A. Gelbukh (Ed.): CICLing 2018, LNCS 13397, pp. 129–141, 2023.

https://doi.org/10.1007/978-3-031-23804-8_11

Table 1. Distribution of emojis in languages’ tweets. (The hearts in the table are of different colors. The most frequent one is *red*, the second most frequent one is *blue* and the last one is *purple heart*)

Lan	😊	❤️	😄	😞	😡	😃	😏	😓	😇	😬	😭	😡	😄	😓	❤️	💙	💜	
Eng	17.1	15.3	10.0	5.7	4.9	4.7	3.8	3.7	3.6	3.5	3.3	3.2	3.1	3.0	2.8	2.7	2.6	
Spa	9.7	10.8	13.1	3.1	2.7	6.5	6.3	6.8	6.0	3.4	6.1	4.0	4.6	5.4	3.4	2.8	3.4	2.0
Hin	9.7	6.8	7.5	3.9	2.8	9.5	4.9	1.5	6.7	2.9	4.3	9.4	8.6	7.7	7.2	1.9	2.3	2.6
Tel	22.7	5.7	16.6	1.4	0.3	10.8	3.9	0.3	3.0	1.6	0.9	13.4	5.1	6.7	4.2	2.3	0.9	0.5

messages and web pages. Originally meaning pictograph, the word *emoji* comes from Japanese e (絵, picture) + moji (文字, character) [20]. Despite immense linguistic diversity, emojis and their definitions remain almost identical across all the major languages. Emojis capture a more mutually shared medium of communication, especially, in case of related cultures.

A frequent usage of social media platforms is microblogging. These microblogs comprise of limited text with an emoji that represents the emotions related to that text. Hence, we establish a general correlation between the text and the emoji, where emoji is the corresponding text’s tag in the microblog. Utilizing this aspect of emojis, we assert that sentences with similar corresponding emojis in different languages carry similar semantic features.

In this paper, we propose a novel unified framework called *Classification of Emojis using Siamese Network Architecture (CESNA)*. CESNA model consists of twin Bi-directional Long Short-Term Memory Recurrent Neural Networks (Bi-LSTM RNN) with shared parameters and a contrastive energy function, based on a similarity metric, joining them. The applied energy function suits discriminative training for energy-based models [14].

CESNA learns the shared model parameters and the similarity metric by minimizing the energy function connecting the twin networks. Parameter sharing and the similarity metric guarantee that, if the emoji of sentences on both the individual Bi-LSTM networks is same, then they are nearer to each other in the emoji space, else they are far away from each other. For example, the representations of “*The Big Bang Theory was funny 😄*” and “*चुनाव मजेदार थे 😄*” (The elections were funny) should be nearer to each other than those of “*The Big Bang Theory was so funny today 😄*” and “*बिग बैंग थ्योरी उबाऊ था 😞*” (Big Bang Theory was boring). The learned similarity metric is used to model the similarity between sentences of different languages into a common emoji space.

The rest of the paper is organized as follows. Section 2 presents the previous approaches to conquer the problem. Section 3 describes the evaluation dataset and Sect. 4 describes the architecture of CESNA. Section 5 explains the training and testing phase of CESNA. Section 6 details the baselines. In Sect. 7, the experimental set-up and results are presented. Finally, Sect. 8 concludes the paper.

2 Related Work

Distributional semantics [16] approach captures the overall sentence’s semantic value but does not maintain information of the words’ order. [17] assigns sentiment polarity to words or phrases. Polarity of its constituents assigns the score to the sentence. The information loss of the words’ sequence leads to the wrong classification. e.g.; In “*I am not happy*”, “*not*” carries a negative sentiment and “*happy*” has a positive sentiment. The combination gives a neutral sentiment, whereas the sentence is truly negative. Bag of n-grams limits the problem but does not eliminate it completely.

BiLSTM model [3] provides a solution to the problem of maintaining the sentence’s sequence, by using recurrent neural network to embed sentences. They propose two types of embeddings based on words and characters. This approach presents an effective model for capturing the content and sequence in the sentence. However, the approach requires immense amount of data to train and hence will fail in case of languages with fewer resources.

Another line of research [2, 12] utilizes rules and vocabulary of the languages to classify sentences. These techniques are highly accurate but susceptible to the problems of spelling errors and improper sentences. And these problems are very frequent in informal texts such as tweets. Also, in case of Hindi, [19] have trained a multinomial naive bayes model on annotated tweets to solve the problem.

Additionally, there have been efforts by researchers [18] to generate more annotated resources by utilizing available raw corpus. They employ the availability of different domains to construct a Multi-arm Active Transfer Learning (MATL) algorithm to label raw samples and continuously add them to the original dataset. Each step updates the algorithm’s parameters using reinforcement learning with a reward function. The above approach works well for the domains considered in their work - sports, movies and politics. A formal grammar and vocabulary structure these domains, whereas, tweets do not follow this trend. Hence, the model is inapplicable to unstructured tweets. The new resources depend on the available resources’ domain, which is risky, especially in the case of tweets that do not comply with any specific domain.

Most of the work done in the fields of emoji prediction and sentiment analysis is on major languages such as English or Spanish. Hence, the assumption in these approaches is the availability of immense data.

Usually, methods that require immutable words are ineffective. Applying languages’ characters instead of words is a better approach. Given their proven effectiveness in [1, 7, 9, 11, 13, 21, 22], we use Bidirectional LSTMs (Bi-LSTMs) based on character n-grams. This approach produces embeddings based on the sequence of character n-grams, thus eliminating the problems of spelling mistakes and agglutination (in the case of some languages such as Telugu).

Although Bi-LSTMs manage mapping of sentences to an emoji space, we also require the distance between the sentences with the similar sentiment to be closer and the sentences with the different sentiment to be farther. For this reason, we use the architecture of siamese networks. This architecture possesses

the capability of learning similarity from the given data without requiring specific information about the classes.

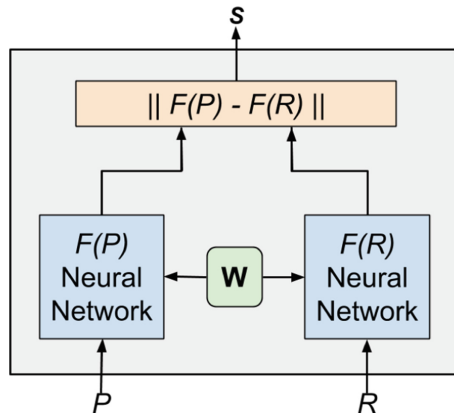


Fig. 1. Siamese Network

2.1 Siamese Networks

[5] introduced siamese neural networks (shown in Fig. 1) to solve the problem of signature verification. Later, [6] applied the architecture with discriminative loss function for face verification. These networks also effectively enhance the quality of visual search [10, 15]. Recently, [8] solved the problem of community question answering applying these networks.

Let $F(X)$ be the family of functions with parameters W . $F(X)$ is differentiable with respect to W . Siamese network seeks a value of the parameter W such that the symmetric similarity metric is small if X_1 and X_2 belong to the same category, and large if they belong to different categories. The scalar energy function $S(R, P)$ that measures the emoji's relatedness between tweets of resource-poor (P) language and resource-rich (R) language can be defined as:

$$S(P, R) = ||F(P) - F(R)|| \quad (1)$$

In CESNA, the network takes tweets from both the languages as input. The loss function is minimized such that $S(P, R)$ is small if the R and P contain the same emoji and large otherwise.

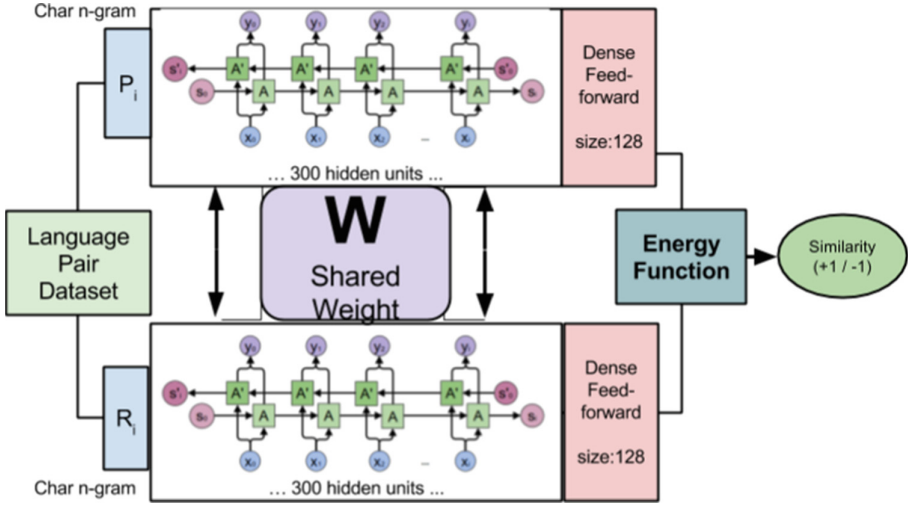


Fig. 2. Architecture of CESNA

3 Dataset Creation

The twitter datasets for different languages are given below:

- **English:** Tweets from the ids given by [3]. The dataset consists of the tweets with 18 most frequent emojis, which is, 500,000 tweets.
- **Spanish:** Tweets containing the most frequent emojis present in English tweets, which is, 100,000 tweets.
- **Hindi:** Tweets containing the most frequent emojis present in English tweets, which is 15000.
- **Telugu:** Tweets containing the most frequent emojis present in English tweets, which is, 6000.

Table 1 demonstrates the distribution of the emojis in the above datasets.

4 Architecture of CESNA

As shown in Fig. 2, CESNA consists of a Bi-LSTM pair and a dense feed forward layer at the top. The Bi-LSTMs capture the sequence and constituents of the sentence and project them to a emoji space. We connect the yielded emoji vectors to a layer that measures similarity between them. The contrastive loss function combines the similarity measure and the label. Back-propagation through time computes the loss function’s gradient with respect to the weights and biases shared by the sub-networks.

Table 2. Number of unique character trigrams and Words in the datasets

Language	Hin	Tel	Eng	Spa
Char trigrams	30849	21453	47924	42261
Words	41731	29298	72182	100171

4.1 Primary Representation

Twitter Data consists of a lot of spelling errors, out-of-vocabulary words and word variations. The way of writing a word may also convey emotions (e.g.; “Hiiii” conveys a positive emotion whereas “Hi” is a neutral emotion). Hence, we use character trigrams to embed the sentence instead of using words. This approach takes care of the spelling errors and out-of-vocabulary words because a partial match exists in the character trigrams. Character trigrams take the information of all the word’s inflections, thus, eliminating the problem of agglutination. This method, also, captures the information of different writing variations. Computational complexity is reduced as the number of words exceeds character trigrams. Table 2 shows the comparison between the number of unique words and unique trigrams in our case. The approach represents a sentence using a vector with number of dimensions equal to the number of unique character trigrams in the training dataset.

We input character-based term vectors of the resource-poor and resource-rich language’s tweets and a label to the twin networks of CESNA. The label indicates whether the samples should be nearer or farther to each other in the emoji space. For positive samples (expected nearer in the emoji space), term vectors of tweets (one from resource-poor and one from resource-rich) with the same emoji are input to the twin networks. For negative samples (expected farther from each other in the emoji space), term vectors of tweets (one from resource-poor and one from resource-rich) with different emojis are input to the twin networks.

4.2 Bi-directional LSTM Network

We map each sentence-pair into $[p_i, r_i]$ such that $p_i \in \mathbb{R}^m$ and $r_i \in \mathbb{R}^n$, where m and n are the total number of character trigrams in the resource-poor language and the resource-rich language respectively.

Bi-LSTM model encodes the sentence twice, one in the original order (forward) of the sentence and one in the reverse order (backward). Back-propagation through time (BPTT) [4] calculates the weights for both the orders independently. The algorithm works in the same way as general back-propagation, except in this case the back-propagation occurs over all the hidden states of the unfolded timesteps.

We apply element-wise Rectified Linear Unit (ReLU) to the output encoding of the BiLSTM. ReLU is defined as: $f(x) = \max(0, x)$. We choose ReLU here because it simplifies back-propagation, causes faster learning and avoids saturation.

The architecture’s final dense feed forward layer converts the output of the ReLU layer into a fixed length vector $s \in \mathbb{R}^d$. In our architecture, we have empirically set the value of d to 128. The overall model is formalized as:

$$s = \max\{0, W[fw, bw] + b\} \quad (2)$$

where W is a learned parameter matrix (weights), fw is the forward LSTM encoding of the sentence, bw is the backward LSTM encoding of the sentence, and b is a bias term, then passed through an element-wise ReLU.

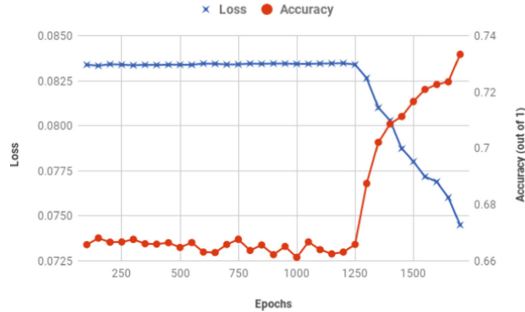


Fig. 3. Loss and Accuracy vs Epochs

5 Training and Testing

We train CESNA on a tweet in resource-poor language with a tweet from resource-rich language to capture the similarity in the tweets’ emojis. CESNA differs from the other deep learning counterparts due to its property of parameter sharing. Training the network with a shared set of parameters not only reduces the number of parameters (thus, save many computations) but also ensures that the sentences of both the languages are project into the same emoji space. We learn the shared network’s parameters with the aim to minimize the distance between the tweets with the same emojis and maximize the distance between the tweets with different emojis.

Given an input p_i, r_i where p_i and r_i are tweets from resource-poor and resource-rich languages respectively and a label $y_i \in \{-1, 1\}$, the loss function is defined as:

$$l(p_i, r_i) = \begin{cases} 1 - \cos(p_i, r_i), & y = 1; \\ \max(0, \cos(p_i, r_i) - m), & y = -1; \end{cases} \quad (3)$$

where m is the margin by which dissimilar pairs should move away from each other. It varies between 0 to 1. We minimize the loss function such that pair of tweets with the label 1 (same emoji) project nearer to each other and pair of tweets with the label -1 (different emoji) project farther from each other in

Table 3. Comparison between different dataset pairs for 5,10,18 emojis (classes). P,R,F1 are Precision, Recall and F-scores of the models respectively.

Dataset Pair	5			10			18		
	P	R	F1	P	R	F1	P	R	F1
Eng-Hin	0.68	0.70	0.69	0.52	0.56	0.54	0.46	0.43	0.44
Eng-Tel	0.63	0.66	0.64	0.48	0.43	0.45	0.42	0.39	0.40
Eng-Spa	0.71	0.72	0.71	0.58	0.59	0.58	0.42	0.42	0.42
Hin-Tel	0.54	0.58	0.56	0.45	0.47	0.46	0.39	0.33	0.35
Eng-Eng	0.74	0.73	0.73	0.62	0.60	0.61	0.49	0.54	0.51

the emoji space. The model trains by minimizing the overall loss function in a batch. The objective is to minimize:

$$L(\Lambda) = \sum_{(p_i, r_i) \in C \cup C'} l(p_i, r_i) \quad (4)$$

where C contains the batch of same emoji tweet pairs and C' contains the batch of different emoji tweet pairs. Back-propagation through time (BPTT) updates the parameters shared by the Bi-LSTM sub-networks.

For testing, we randomly sample a certain number (100 in our case) of tweets for each emoji R_{emoji} from the language corpus with higher amount of data. For every input, we then apply the trained model to get the similarity between the input and all corresponding R_{emoji} . The R_{emoji} with the most number of matches with the input is finally selected as the correct emoji.

In case the testing data of both resource-rich and resource-poor languages do not contain emojis, we use the abundant resources of one language to construct a state-of-the-art emoji prediction model [3] and then utilize it to aid the resource-poor language’s prediction.

6 Baselines

The approaches vary based on the language in consideration. Hence, we accordingly define the baselines below. English, Japanese and Spanish enjoy the highest share of data on Twitter¹. We consider English and Spanish because of their script and typological similarity (both are Subject-Verb-Object). The baselines considered for resource-rich languages are:

- **Average Skip-Gram Vectors (ASV):** We train a Word2Vec skip-gram model [16] on a corpus of 65 million raw (unannotated) tweets in English and 20 million raw tweets in Spanish. Word2Vec provides a vector for each word.

¹ The Many Tongues of Twitter - MIT Technology Review.

Table 4. Comparison with the baselines. ASV is Average Skipgram Vectors, Bi-LSTM (W) and Bi-LSTM (C) refer to Word and Character based Bi-LSTM models. They are baselines for English and compare to CESNA (Eng-Eng). DSC-T is Domain Specific Classifier for Telugu and compares to CESNA (Tel-Eng). MNB-H refers to Multinomial Bayes Model for Hindi and compares to CESNA (Hin-Eng). P,R,F1 are the Precision, Recall and F-scores respectively.

Dataset Pair	5			10			18		
	P	R	F1	P	R	F1	P	R	F1
ASV	0.59	0.60	0.59	0.44	0.47	0.45	0.32	0.34	0.35
Bi-LSTM (W)	0.61	0.61	0.61	0.45	0.45	0.45	0.34	0.36	0.35
Bi-LSTM (C)	0.63	0.63	0.63	0.48	0.47	0.47	0.42	0.39	0.40
DSC-T (RF)	0.34	0.35	0.34	0.31	0.32	0.31	0.24	0.23	0.23
MNB-H	0.45	0.49	0.46	0.42	0.43	0.42	0.38	0.36	0.37
CESNA (Eng-Eng)	0.74	0.73	0.73	0.62	0.60	0.61	0.49	0.54	0.51
CESNA (Hin-Eng)	0.68	0.70	0.69	0.52	0.56	0.54	0.46	0.43	0.44
CESNA (Tel-Eng)	0.63	0.66	0.64	0.49	0.47	0.48	0.41	0.44	0.42

We average the words’ vectors in the tweet to get the vector for the sentence. So, each sentence vector is defined as:

$$V_s = \frac{\sum_{w \in W_s} V_w}{|W_s|} \quad (5)$$

where V_s is vector of the sentence s , W_s is the set of words and V_w is the vector of word w . After obtaining each message’s embedding, we train an L2-regularized logistic regression, (with ϵ equal to 0.001).

- **Bidirectional LSTM (Bi-LSTM):** There are two approaches - word based and character based Bi-LSTM embeddings. We model the architecture as described in [3]. We use the same design as a part of our model, which is explained in Sect. 4.2.

Hindi and Telugu are the 3rd and 17th most spoken language in the world respectively. But they hold a relatively low share of twitter data. The major reason is that the speakers of Hindi and Telugu on Twitter primarily use the transliterated form of their respective language. The baselines for these languages are:

- **Domain Specific Classifier (Telugu) (DSC-T):** We train a Word2Vec model on a corpus of 700,000 raw Telugu sentences provided by Indian Languages Corpora Initiative (ILCI). We train a Random Forest (RF) and Support Vector Machines (SVM) classifier (given by [17]) on the Telugu Twitter dataset to structure our baseline for Telugu language.
- **Multinomial Naive Bayes (Hindi) (MNB-H):** We train a multinomial naive bayes model (given by [19]) on the Hindi Tweets dataset to form our baseline for Hindi language.

7 Experiments and Evaluation

In order to study the comparison between CESNA and the previous models, we performed an array of experiments. In the first experiment (Sect. 7.1), we analyze the model for varying language pairs and make a comparison between them. In the second experiment (Sect. 7.2), we compare our model against previous approaches in the problem. In the third experiment (Sect. 7.3), we train our architecture on clusters of emojis instead of unique ones.

7.1 Experiments for Different Language Pairs

The experiment is a classification task. We take the English and Hindi Twitter datasets (Eng-Hin) and align each Hindi tweet with English tweets of the same emoji (positive samples) and label them 1. Similarly, we also randomly sample the same number of English tweets with different emoji (negative samples) for each Hindi tweet and label them -1.

We perform the experiment thrice taking 5 most frequent emojis (5 classes), 10 most frequent emojis (10 classes) and all the emojis (18 classes) in Hindi. Similarly, we repeat the experiment for English-Telugu (Eng-Tel) dataset pair, English-Spanish (Eng-Spa) dataset pair, English-English (Eng-Eng) dataset pair and Hindi-Telugu (Hin-Tel) dataset pair, taking 5 most frequent emojis (5 classes), 10 most frequent emojis (10 classes) and all the emojis in the language with lesser resource respectively for each case. The results of the experiments are given in Table 3.

7.2 Comparison with the Baselines

In this experiment, we compare our model against the baselines (defined in Sect. 6). We defined the baselines for resource-rich languages on English. So, we perform contrastive learning of our model using data made by aligning each English tweet with a set of positive English tweet samples (with the same emoji) with label 1 and a set of negative English tweet samples (with different emoji) of the same size with label -1.

In the case of resource-poor languages, i.e. Hindi and Telugu, we perform contrastive learning of our model using data made by aligning each of the resource-poor language (Hindi and Telugu) tweet with a set of positive English tweet samples (with the same emoji) with label 1 and a set of negative English tweet samples (with different emoji) of the same size with label -1.

7.3 Clustering Based Approach with CESNA

In our previous experiment (Sect. 7.1), we observed that in several test scenarios the tweet is incorrectly classified to its nearest neighbor in the semantic space of emojis. We observe that the emojis form clusters in the semantic map. These clusters reduce multiple unique emojis to a single class for this experiment. So, we finally arrive at three clusters (Table 5).

Table 5. Clustering the emojis to a single emoji³ The emojis in the *heart* cluster are of different colors. The first one is *red*, second one is *purple* and third one is *blue* in color.



Table 6. Results after clustering the Emojis

Language Pair	P	R	F1
CESNA (Eng-Eng)	0.83	0.85	0.83
CESNA (Hin-Eng)	0.80	0.79	0.79
CESNA (Tel-Eng)	0.72	0.74	0.73

7.4 Evaluation of the Experiments

We observe from Table 3 that the best overall results for multilingual emoji classification is the English-Spanish pair. This is due to the English-Spanish pair containing the maximum number of tweet pairs. We also note from Fig. 3 that with increasing number of epochs, the accuracy and overall performance considerably increases.

We also find that multiple times a tweet classifies into a related class. e.g.; A tweet of class (*purple heart emoji*) is classified into a more frequent emoji (*red heart emoji*). To verify this behavior, we conducted another experiment in Sect. 7.3 to approach this from the perspective of emojis’ clustered classes. The results (given in Table 6) demonstrate that fewer classes lead to better accuracy. This reduction in the number leads to a more even distribution of classes in the data. The drawback of this approach, though, is the loss of information about emojis. Hence, it only benefits when such data loss is acceptable.

From Table 4, we observe that CESNA outperforms the state-of-the-art approaches significantly, especially in the case of resource-poor languages. Interestingly, Table 4 also shows that using shared parameters (Siamese Networks) instead of a single Bi-LSTM network leads to an improvement in performance. CESNA learns representation, specifically, for the task of emoji-based classification. It also leverages the relatively resource-rich language for the improvement in the resource-poor language’s accuracy.

8 Conclusions

In this paper, we proposed CESNA for emoji prediction of resource-poor languages which solves the problem by projecting the resource-poor language and resource-rich language in the same emoji space. CESNA employs twin Bidirectional LSTM networks with shared parameters to capture an emoji-based representation of the sentences. These emoji-based representations in conjunction with a similarity metric group sentences with similar emoji together.

A clustering based approach used in conjunction with CESNA boosts the performance of overall emoji prediction further. Experiments conducted on different Twitter datasets revealed that CESNA outperforms the current state-of-the-art approaches significantly.

In future, we would apply the current model on more applications based on learning similarity like question-answering, conversation systems and semantic similarity. Though, of course, the presence and impact of emojis would be limited in other areas. Also, we believe that there is a good case for integration of attention-based models in the subnetworks.

References

1. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint [arXiv:1409.0473](https://arxiv.org/abs/1409.0473) (2014)
2. Balamurali, A., Joshi, A., Bhattacharyya, P.: Cross-lingual sentiment analysis for indian languages using linked wordnets. In: Proceedings of COLING 2012: Posters, pp. 73–82 (2012)
3. Barbieri, F., Ballesteros, M., Saggion, H.: Are emojis predictable? arXiv preprint [arXiv:1702.07285](https://arxiv.org/abs/1702.07285) (2017)
4. Boden, M.: A guide to recurrent neural networks and backpropagation. the Dallas project (2002)
5. Bromley, J., Guyon, I., LeCun, Y., Säcker, E., Shah, R.: Signature verification using a” siamese” time delay neural network. In: Advances in Neural Information Processing Systems, pp. 737–744 (1994)
6. Chopra, S., Hadsell, R., LeCun, Y.: Learning a similarity metric discriminatively, with application to face verification. In: Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on. vol. 1, pp. 539–546. IEEE (2005)
7. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint [arXiv:1412.3555](https://arxiv.org/abs/1412.3555) (2014)
8. Das, A., Yenala, H., Chinnakotla, M., Shrivastava, M.: Together we stand: Siamese networks for similar question retrieval. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). vol. 1, pp. 378–387 (2016)
9. Dhingra, B., Zhou, Z., Fitzpatrick, D., Muehl, M., Cohen, W.W.: Tweet2vec: Character-based distributed representations for social media. arXiv preprint [arXiv:1605.03481](https://arxiv.org/abs/1605.03481) (2016)
10. Ding, S., Cong, G., Lin, C.Y., Zhu, X.: Using conditional random fields to extract contexts and answers of questions from online forums. In: ACL. vol. 8, pp. 710–718 (2008)
11. Dyer, C., Ballesteros, M., Ling, W., Matthews, A., Smith, N.A.: Transition-based dependency parsing with stack long short-term memory. arXiv preprint [arXiv:1505.08075](https://arxiv.org/abs/1505.08075) (2015)
12. Joshi, A., Balamurali, A., Bhattacharyya, P.: A fall-back strategy for sentiment analysis in hindi: a case study. In: Proceedings of the 8th ICON (2010)
13. Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., Dyer, C.: Neural architectures for named entity recognition. arXiv preprint [arXiv:1603.01360](https://arxiv.org/abs/1603.01360) (2016)
14. LeCun, Y., Huang, F.J.: Loss functions for discriminative training of energy-based models. In: AISTats (2005)
15. Liu, Y., Li, S., Cao, Y., Lin, C.Y., Han, D., Yu, Y.: Understanding and summarizing answers in community-based question answering services. In: Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1, pp. 497–504. Association for Computational Linguistics (2008)

16. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: *Advances in neural information processing systems*, pp. 3111–3119 (2013)
17. Mukku, S.S., Choudhary, N., Mamidi, R.: Enhanced sentiment classification of telugu text using ml techniques. In: *SAaip@ IJCAI*. pp. 29–34 (2016)
18. Mukku, Sandeep Sricharan, Oota, Subba Reddy, Mamidi, Radhika: Tag me a label with multi-arm: active learning for Telugu sentiment analysis. In: Bellatreche, Ladjel, Chakravarthy, Sharma (eds.) *DaWaK 2017*. LNCS, vol. 10440, pp. 355–367. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-64283-3_26
19. Sarkar, Kamal, Chakraborty, Saikat: A sentiment analysis system for indian language tweets. In: Prasath, Rajendra, Vuppala, Anil Kumar, Kathirvalavakumar, T.. (eds.) *MIKE 2015*. LNCS (LNAI), vol. 9468, pp. 694–702. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-26832-3_66
20. Taggart, C.: *New Words for Old: Recycling Our Language for the Modern World*. Michael O’Mara Books (2015)
21. Vinyals, O., Kaiser, L., Koo, T., Petrov, S., Sutskever, I., Hinton, G.: Grammar as a foreign language. In: *Advances in Neural Information Processing Systems*, pp. 2773–2781 (2015)
22. Wang, P., Qian, Y., Soong, F.K., He, L., Zhao, H.: Learning distributed word representations for bidirectional lstm recurrent neural network. In: *HLT-NAACL*, pp. 527–533 (2016)



Aspect-Sentiment Embeddings for Company Profiling and Employee Opinion Mining

Rajiv Bajpai¹, Devamanyu Hazarika^{2(✉)}, Kunal Singh⁴, Sruthi Gorantla⁵,
Erik Cambria³, and Roger Zimmermann²

¹ Accenture, Pune, India

² National University of Singapore, Singapore, Singapore
{hazarika, rogerz}@comp.nus.edu.sg

³ Nanyang Technological University, Singapore, Singapore
cambria@ntu.edu.sg

⁴ Indian Institute of Technology Kharagpur, Kharagpur, India
kunal.singh@iitkgp.ac.in

⁵ Indian Institute of Science, Bangalore, Bengaluru, India
gorantlas@iisc.ac.in

Abstract. With the multitude of companies and organizations around today, ranking them and choosing one out of the many is a difficult and cumbersome task. Although there are many available metrics that rank companies, there is an inherent need for a generalized metric that takes into account the different aspects that constitute employee opinions of the companies. In this work, we aim to overcome the aforementioned problem by generating aspect-sentiment based embedding for the companies by looking into reliable employee reviews of them. We created a comprehensive dataset of company reviews from the famous website [Glassdoor.com](https://www.glassdoor.com) and employed a novel ensemble approach to perform aspect-level sentiment analysis. Although a relevant amount of work has been done on reviews centered on subjects like movies, music, etc., this work is the first of its kind. We also provide several insights from the collated embeddings, thus helping users gain a better understanding of their options as well as select companies using customized preferences.

Keywords: Aspect-based sentiment analysis · Sentiment embedding · Company profiling · Extreme Learning Machine

1 Introduction

Emotions, sentiment, and judgments on the scale of good—bad, desirable—undesirable, approval—disapproval are essential for human-to-human communication. Understanding human emotions, deciphering humans' emotional reasoning and how humans express them in their language is key to enhancing human-machine interaction. In this era of social media, the WWW provides

new tools that create and share ideas and opinions with everyone efficiently. Capturing public opinion on social media about events, political movements or any other topics bears a potential for interest amongst the scientific community. That is mainly because of two reasons - firstly, these opinions can help individuals in their decision making process. Secondly, organizations will utilize such data in order to glean public opinion regarding services and products so as to fine-tune/develop their business strategies.

Sentiment analysis is the study of opinions and sentiments from content that spans from the unimodal to the multimodal [28–30]. Recent approaches to sentiment analysis have focused on the use of linguistic patterns and deep neural networks. The ability to identify aspects within texts is also equally important. An aspect is defined as the product feature; for instance, in the sentence “the battery lasts long”, *battery* is the aspect for which positive sentiment is expressed. The main challenge of sentiment analysis is identifying aspects and their corresponding sentiment. In our case, we do so by merging linguistic patterns and an ELM classifier.

Employees are organizational assets and their opinion plays a vital role in any organization’s growth. Job Search Engines and review websites have evolved to become an ocean of employee reviews. Employee reviews play a vital role for a company’s growth as it improves the relationship between management and the employees via improving staff welfare and morale. These reviews also help prospective employees in selecting a company that meets their criterion. Despite such reviews being important data sources for sentiment mining, they have failed to draw the attention of the scientific community. To the best of our knowledge, only the work of [21] utilized company reviews from Glassdoor. However, even their work was limited to extracting only topics and sentiments from the reviews. In this work, we built a large dataset of employee reviews of companies in Singapore sourced from Glassdoor. Different types of analysis, e.g., aspect extraction, aspect based sentiment analysis were then carried out on this dataset by blending ELM with sentic patterns. To this end, we developed representational embeddings of the companies based on the sentiment score of various different aspects of the companies. In particular, each company is represented in a 30 dimensional space where each dimension corresponds to the average sentiment score of an aspect. Some of these aspects are ‘company culture’, ‘salary’, ‘location’, etc.

In this paper, we used Glassdoor as our source for the preparation of the dataset comprising of 40k reviews. The volume of reviews span a diverse range of aspects (positive and negative) that describe the company based on personal opinions of the reviewers. There are two main contributions of this paper:

- Creation of a large dataset derived from Glassdoor for aspect level sentiment analysis. This dataset contains the reviews of employees working or who previously worked at the corresponding companies.
- Introducing aspect-sentiment embeddings of the companies in order to find similarities between companies in the similar or differing sectors. Aspect-sentiment embeddings project each company onto an n-dimensional space where each dimension corresponds with the overall aspect-sentiment strength

of the employees. Aspects are different features of a company, e.g., *salary*, *location*, *work-life balance*, etc. This is particularly useful for job seekers who are looking to find companies that suit their preferences.

The rest of the paper is organized as follows: Sect. 1 discusses sentiment analysis literature; collection and preparation of the dataset are featured in Sect. 2; we discuss the algorithm details in Sect. 4; experimental results of this study are presented in Sect. 5; finally, Sect. 6 concludes the paper.

Related Works

Identifying emotions associated with employers is just one of the many possible applications of sentiment analysis. We could also analyze industries or professions as a whole, or consider the relationship between the emotional content of reviews with the corresponding salaries of employees. One would expect higher salaries to correlate with more positive emotions, but we might also see an inverse correlation in some cases, perhaps indicating the use of ‘golden handcuffs’.

Sentiment analysis systems can be broadly categorized into knowledge-based [6] or statistics-based systems [8]. Initially, knowledge bases were more commonly used for the identification of emotions and polarity in text. However, at present, sentiment analysis researchers more commonly use statistics-based approaches, with a specific focus on supervised statistical methods. For example, Pang et al. [24] compared the performance of different machine learning algorithms on a movie review dataset: using a large number of textual features, they obtained 82.90% accuracy.

Other unsupervised or knowledge-based approaches to sentiment analysis include Turney et al. [31], which used seed words to calculate the polarity and semantic orientation of phrases, as well as Melville et al. [13] which proposed a mathematical model to extract emotional clues from blogs and then used the information for sentiment detection.

Sentiment analysis research can also be categorized as single-domain [24] versus cross-domain [2]. The work presented in [23] discusses the use of spectral feature alignment to: 1) group domain-specific words from different domains into clusters, and 2) reduce the gap between domain-specific words of two domains using domain independent words. Bollegala et al. [3] developed a sentiment-sensitive distributional thesaurus by using labeled training data from source domain and unlabeled training data from both source and target domains. Some recent approaches [9, 22] used SentiWordNet [1], a very large sentiment lexicon developed by automatically assigning polarity value to WordNet [20] synsets. In SentiWordNet, each synset has three sentiment scores along three sentiment dimensions: positivity, negativity, and objectivity.

As discussed in the introduction, there are hardly any works on mining opinions from company reviews written by employees. Moniz et al. [21] proposed an aspect-sentiment model based on the Latent Dirichlet Allocation (LDA). According to their study, the results of the articulate aspect-polarity model showed that it might be advantageous for investors to combine an appraisal of employee satisfaction with other existing methods for forecasting firm earnings. The research explained and analyzed the sentiments of a stakeholder group which is possibly

neglected: the firm’s employees. The researchers initially used online employee reviews in order to capture employee satisfaction and utilized LDA to consider salient aspects in employees’ reviews. From that, they manually derived a latent topic that appeared to be associated with the firm’s outlook. Secondly, they created an entire document by grouping employee reviews for each firm, and using the General Inquirer dictionary to count positive and negative terms, they measured sentiment as the polarity of the composite document. Their model suggested that employee satisfaction could be formulated as a function of the firm’s outlook and employee sentiment.

2 Dataset Collection

In our research, we used the popular job recruiting site *Glassdoor.com* as our source to prepare the dataset. The website provides tons of reliable reviews for many companies written mostly by employees, ex-employees or directly associated clients. The content of the reviews possess different aspects (positive and negative) that represent the company from the individual perspectives of the writers. The anonymity of writers enhances the authenticity of the review, thus, this site was our primary source for the dataset collection. The language used in the reviews was found to be highly formal with very minimal usage of slang, decreasing the effort required for data cleaning, normalization, and tokenization. These ‘clean’ words had a high match rate with the dictionary we used [11] for creating the word embeddings, thus improving the performance of the ELM model used in our ensemble network.

The review structure in *Glassdoor.com* consists of a general description followed by both *pros* and *cons* to be written and listed by the writer. This rigid structure aids us in building a balanced dataset comprising of both positive and negative reviews associated with the companies. However, one must be wary of comments like “I really don’t have anything to say/complain about here” in the pros/cons section, which we believe to represent false positives or false negatives respectively. We decided to include these comments in our dataset to represent real-world instances where such false comments are prevalent.

The dataset has been collected by using the official API¹ of Glassdoor. We created a diverse list of 60 well-known companies representing various domains such as technology, finance, energy, hospitality, etc. Finally, we collected a total of 20,000 reviews for all companies. As mentioned earlier, the reviews listed both the pros and cons about the company that is reviewed. Given this sophistication, it was easier for us to split each review into two sub-reviews containing positive and negative opinions respectively. This in turn enabled the automatic labeling of said reviews. Despite doing so, we were careful to manually check the labeling afterwards in order to filter out wrong labels.

Here is an excerpt from one of the positive reviews written for *Accenture*—“*They have great career opportunities, a never ending supply of interesting work, competitive compensation, wonderful benefits, great people, wonderful training*”

¹ <https://www.glassdoor.com/developer/index.htm>.

programs, a tremendous number of brilliant professionals in their fields ready to help, and great core values". This review, like others, is full of important aspects such as *opportunities, compensation, benefits, etc.*, which provides extensive opinions on different facets/characteristics of the company and thus allow our team to prepare a comprehensive review dataset.

Table 1. Number of reviews per company

Company	Reviews	Company	Reviews
Accenture	1000	HP	150
Adobe	998	HSBC Holdings	1850
Aeropostale	874	IBM	150
Aflac	368	Intel Corporation	998
Autodesk	752	Intuit	972
Bank of China	212	Marriot International	980
Booz Allen Hamilton	976	Microsoft	1000
Broadcom	151	Mosanto	396
Brocade	990	Morningstar	733
Camden Property	203	National Instruments	712
Capital One	997	NetApp	800
CarMax	959	Nordstorm	839
Chesapeake Energy	725	OCBC	268
Cisco	980	Paychex	916
Citibank	1852	Qualcomm	919
Colgate-Palmolive	776	Quest Global	150
Creative Technology	150	Rackspace Hosting	732
Darden Restaurants	994	Samsung	150
DBS	288	SCB	942
Devon Energy	336	Singtel	480
DreamWorks Animation	978	StarBucks	828
EOG Resources	127	Starhub	150
FactSet	976	Stryker	904
FedEx Corporation	850	SVB Financial Software	277
Flextronics	150	J.M. Smucker Company	318
General Mills	1036	Ultimate Software	524
Goldman Sachs	970	Umpqua Bank	242
Google	756	Union Overseas Bank	265
Hasbro	458	World Foods Market	757
Herman Miller	540	Yes Bank	176

In Table 1, we show the number of reviews per company. The aim was to collect 500–1000 most helpful reviews². However, for some companies the number of reviews available on Glassdoor numbered less than 500.

² Reviews for each company in the dataset were selected based on them bearing the most ‘helpful’ review tag provided by [Glassdoor.com](https://www.glassdoor.com).

2.1 Preprocessing

As mentioned earlier, the reviews follow a rigid outline structure regardless of writers. Thus, we shuffled the dataset using a pseudo-random generator so as to break any kind of patterns embedded in the dataset. For the purposes of processing the text, we removed any urls, links and hashtags with the use of regular expressions. However, we retained the smileys and emoticons used in the reviews and included them in our vocabulary so as to exploit the emotional and sentimental hints present in them. As we used a context-based algorithm to create our aspect-sentiment embeddings, retaining these special, non-verbal ‘words’ held a key importance in the performance of the ELM classification. Following this, we used the *NLTK Tokenize Package* to tokenize the reviews into sentences and, finally, into words so as to build up our model’s vocabulary.

3 Backgrounds

3.1 Aspect-Sentiment Embeddings

In this paper, we introduce Aspect-sentiment Embeddings, which projects companies onto an n -dimensional space. In particular, each company is given a sentiment strength for each aspect (e.g., salary, work life, location) based on the opinions mined from employee reviews of the company. In mathematical notation we can say, (s_1, s_2, \dots, s_n) is a vector where s_i is the sentiment score for aspect i and there are a total of n aspects. We constructed such vectors for every company in our dataset, which gave us aspect-sentiment embeddings of the companies.

3.2 Doc2vec for Review Level Embeddings

As we use ensemble architecture to prepare our aspect-sentiment embeddings, the ELM module plays a crucial role as one of the dual paths in the architectural model. To use the ELM module, we need to convert the raw text into review-level summarized embeddings. In our work, we use *Doc2vec* [18] to achieve this task. *Doc2vec*, also known as *paragraph2vec*, is a modification of the *word2vec* algorithm. *Word2vec* itself is a famous algorithm for word embeddings provided by [19] which trains a neural network to extract contextual-based word embeddings based on the CBOW architecture. Such training has been done on a 100 billion words corpus from Google News and the vectors formed are of 300 dimensionality. In contrast, *Doc2vec* is an unsupervised learning of continuous representations for larger blocks of text, such as sentences, paragraphs or entire documents. As the reviews in our dataset are very detailed, employing the *Doc2vec* algorithm is justified. We used the python implementation provided by *gensim*³ to extract 300 dimensional embeddings to be fed into the ELM model for sentimental analysis and classification.

³ <https://radimrehurek.com/gensim/>.

3.3 Sentiment Dictionary/Lexicons

In order to assign aspect polarity score, we created a dictionary of terms along with their polarities to be used by our ensemble algorithm. We used two primary resources, SentiWordNet [11] and SenticNet [5] to create this dictionary. To filter out irrelevant words that act as noise and would thus fail to provide good polarity to the aspects, we kept an absolute threshold of 0.25 in the polarity scores of the terms in both resources. In order to avoid redundancy, when a particular term is present in both SentiWordNet and SenticNet, we gave priority to SentiWordNet and chose the term polarity pair from there. Once the dictionary was created, we used it as a reference lookup table in our algorithm mentioned below.

3.4 Aspect-Level Sentiment Analysis

In opinion mining, different levels of analysis granularity have been proposed, with each having its own advantages and drawbacks [7]. Aspect-based opinion mining [10,12] focuses on the relations between aspects and document polarity. An aspect, also known as an opinion target, is a concept in which the opinion is expressed in the given document. For example, the sentence, “The screen of my phone is really nice and its resolution is superb” contains positive polarity for a phone review, i.e., the author likes the phone. However, more specifically, the positive opinion is about its screen and resolution; these concepts are called opinion targets, or, aspects of this opinion. The task of identifying the aspects in a given opinionated text is called aspect extraction.

There are two types of aspects defined in aspect-based opinion mining: explicit aspects and implicit aspects. Explicit aspects are words in the opinionated document that explicitly denote the opinion target. For instance, in the aforementioned example, the opinion targets ‘screen’ and ‘resolution’ are explicitly mentioned in the text. In contrast, an implicit aspect is a concept that represents the opinion target of an opinionated document, but which is not specified explicitly in the text. One can infer that the sentence, “This camera is sleek and very affordable” implicitly contains a positive opinion of the aspects ‘appearance’ and ‘price’ of the entity camera. These same aspects would be explicit in an equivalent sentence: “The appearance of this camera is sleek and its price is very affordable”.

3.5 Extreme Learning Machines

For classification, we used ELM as a supervised classifier. The ELM approach [4,16] was introduced to overcome some issues in back-propagation network [27] training, specifically, potentially slow convergence rates, the critical tuning of optimization parameters [32], and the presence of local minima that call for multi-start and re-training strategies. The ELM learning problem settings require a training set, X , of N labeled pairs, using the equation (\mathbf{x}_i, y_i) , where $\mathbf{x}_i \in \mathcal{R}^m$ is the i -th input vector and $y_i \in \mathcal{R}$ is the associate expected ‘target’ value; using a scalar output implies that the network has one output unit, without loss of generality.

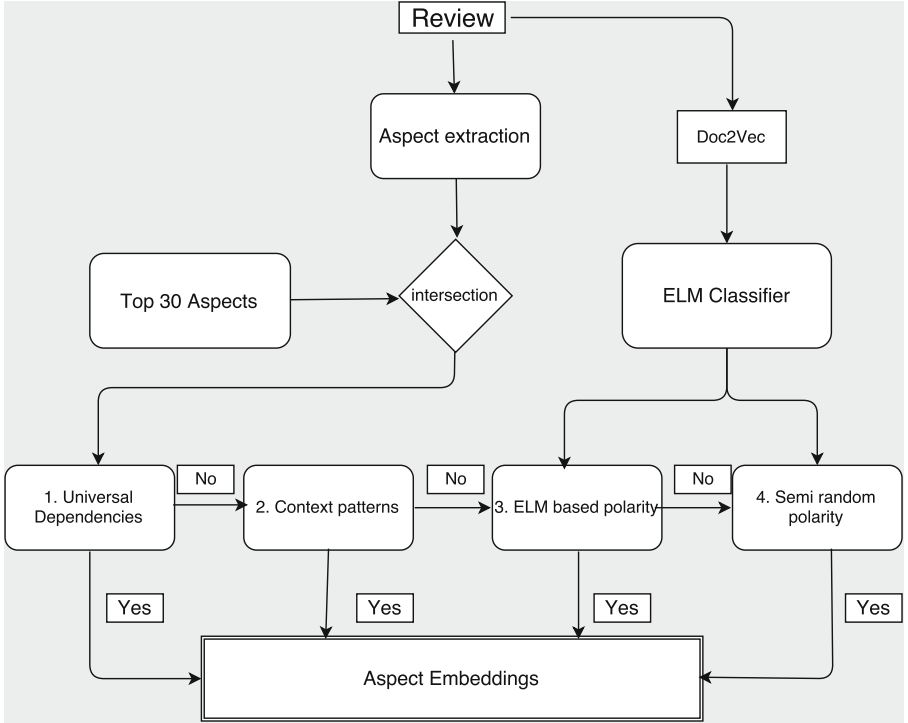


Fig. 1. The flowchart of the algorithm.

The input layer has m neurons and connects to the ‘hidden’ layer (having N_h neurons) through a set of weights $\{\hat{\mathbf{w}}_j \in \mathcal{R}^m; j = 1, \dots, N_h\}$. The j -th hidden neuron embeds a bias term, \hat{b}_j , and a nonlinear ‘activation’ function, $\varphi(\cdot)$; thus the neuron’s response to an input stimulus, \mathbf{x} , is:

$$a_j(\mathbf{x}) = \varphi(\hat{\mathbf{w}}_j \cdot \mathbf{x} + \hat{b}_j) \quad (1)$$

Note that (1) can be further generalized to a wider class of functions [15] but for the subsequent analysis this aspect is not relevant. A vector of weighted links, $\bar{\mathbf{w}}_j \in \mathcal{R}^{N_h}$, connects hidden neurons to the output neuron without any bias [14]. The overall output function, $f(\mathbf{x})$, of the network is:

$$f(\mathbf{x}) = \sum_{j=1}^{N_h} \bar{\mathbf{w}}_j a_j(\mathbf{x}) \quad (2)$$

It is convenient to define an ‘activation matrix’, \mathbf{H} , such that the entry $\{h_{ij} \in \mathbf{H}; i = 1, \dots, N; j = 1, \dots, N_h\}$ is the activation value of the j -th hidden neuron for the i -th input pattern. The \mathbf{H} matrix is:

$$\mathbf{H} \equiv \begin{bmatrix} \varphi(\hat{\mathbf{w}}_1 \cdot \mathbf{x}_1 + \hat{b}_1) & \cdots & \varphi(\hat{\mathbf{w}}_{N_h} \cdot \mathbf{x}_1 + \hat{b}_{N_h}) \\ \vdots & \ddots & \vdots \\ \varphi(\hat{\mathbf{w}}_1 \cdot \mathbf{x}_N + \hat{b}_1) & \cdots & \varphi(\hat{\mathbf{w}}_{N_h} \cdot \mathbf{x}_N + \hat{b}_{N_h}) \end{bmatrix} \quad (3)$$

In the ELM model, the quantities $\{\hat{\mathbf{w}}_j, \hat{b}_j\}$ in (1) are set randomly and are not subject to any adjustment, and the quantities $\{\bar{\mathbf{w}}_j, \bar{b}\}$ in (2) are the only degrees of freedom. The training problem reduces to the minimization of the convex cost:

$$\min_{\{\bar{\mathbf{w}}, \bar{b}\}} \|\mathbf{H}\bar{\mathbf{w}} - \mathbf{y}\|^2 \quad (4)$$

A matrix pseudo-inversion yields the unique L_2 solution, as proven in [16]:

$$\bar{\mathbf{w}} = \mathbf{H}^+ \mathbf{y} \quad (5)$$

The simple, efficient procedure to train an ELM therefore involves the following steps:

1. Randomly set the input weights $\hat{\mathbf{w}}_i$ and bias \hat{b}_i for each hidden neuron;
2. Compute the activation matrix, \mathbf{H} , as per (3);
3. Compute the output weights by solving a pseudo-inverse problem as per (5).

Despite the apparent simplicity of the ELM approach, the crucial result is that even random weights in the hidden layer endow a network with a notable representation ability [16]. Moreover, the theory derived in [17] proves that regularization strategies can further improve its generalization performance. As a result, the cost function (4) is augmented by an L_2 regularization factor as follows:

$$\min_{\bar{\mathbf{w}}} \{\|\mathbf{H}\bar{\mathbf{w}} - \mathbf{y}\|^2 + \lambda \|\bar{\mathbf{w}}\|^2\} \quad (6)$$

4 Detailed Algorithm: Ensemble Architecture

We propose a hybrid algorithm, which works as an ensemble of Unsupervised and Machine Learning approaches, for assigning sentiment labels to the reviews. First, based on the dependency structure of a review we assign polarity to the aspects present in the reviews. This process assumes that the aspect word is connected to a word that is polar and present in the sentiment lexicon. If there is no polar word found connected to the aspect word, according to the sentiment dictionary, we resort to the use of supervised classifier, i.e., ELM. The usage of ELM has multiple benefits like comparable or better performance than other machine learning models like SVMs, and most importantly boasts a significant reduction in model building time. Training time is an important aspect in our work given its high probability to be adapted into an online and real-time application. The flowchart of the proposed algorithm is shown in Fig. 1.

4.1 Aspect Extraction

We used the aspect extraction method by Poria et al. [26], who proposed a hybrid classifier that uses a Convolutional Neural Network for aspect extraction, as well as linguistic patterns for the purposes of pruning the aspect extraction process. The extracted aspects are given below:

- **Job**, is an umbrella aspect that represents the overall characteristic and nature of the job at that particular company, e.g., *Stable and secure job, good people*
- **Employees/Co-workers**, represents the quality of employees, co workers and the company’s relationship with them, e.g., *Very healthy organization with a high-performance culture and very talented employees.*
- **Working time**, clubs aspects of diverse meanings ranging from ‘extra-time’ to ‘time-off’ which signify work hours and trends, e.g., *Great people, very generous vacation and time off including sabbaticals every 5 years.*
- **Management**, explores the managerial aspects of the company, e.g., *Great place to work. Inclusive management process. Great products.*
- **Office culture**, summarizes the office environment and the working style, e.g., *Strong focus on procedures, policies, culture, and people. Great benefits.*
- **Location**, represents the comments on location of the company, e.g., *Competitive salary, Nice location, Full freedom.*
- **Work life**, speaks in particular about the type and quality of work, along with the work-life balance present in the company, e.g., *Great office space and location, interesting products to work on.*
- **Salary**, provides information on the salary margins of the company, e.g., *Salary is OK Bonus is good unless there is a food fight. Too laid back (leads to no innovation).*
- **Perks/Benefits**, compensations, bonuses and miscellaneous benefits are included in this aspect, e.g., *Good perks for this type of job - and vary even across levels of employment. Fitness reimbursement, stock options, sabbaticals, etc.*
- **Job opportunities**, scope of the job in the company, e.g., *Strong culture, good reputation, interesting opportunities, management cares about the careers of the employees they are managing.*
- **Employee experience**, lists the sentiments of employees with different degrees of experience and also the quality of experience to be acquired in the company, e.g., *The size of the org can make it difficult for individuals to have their voices heard, especially for new hires, regardless of their experience.*
- **Official staff**, talks about the strength, quality and hospitality of the staff in the company, e.g., *Hours, upper management, lack of staff.*
- **Job training**, expresses the training frameworks and opportunities provided by the company, e.g., *Inconsistent hours, sometimes no hours. No proper training.*
- **Personal growth**, the possibility of technological and experiential growth for the employee, e.g., *Need patience to sense growth since it is a challenging business and changing company.*

- **Leadership**, discusses the role and efficacy of the leadership/senior officials in the areas of motivation and leadership skills. *Leadership does not know how to utilize experienced, professional talent*
- **Politics**, represents the interaction between people for power, e.g., *Politics drive people for more power.*
- **Company business**, explores the business aspects such as performance, trade, etc. of the company, e.g., *Business of the company is booming.*
- **Career development**, forecasts the future of the individuals and company, e.g., *International job within 2 years.*
- **Vacation**, represents the number of holidays the company provides its staff with, e.g., *Paid vacation for the summer. Free travel within the same country.*
- **Company support**, summarizes the quality of interaction between employees, e.g., *Supervisors take care of their employees.*
- **Flexibility**, represents how flexible the company's environment is, e.g., *Full freedom, work from home allowed.*
- **Performance**, speaks about the type and quality of work done by the employees, e.g., *Extraordinary skills shown by the employees.*
- **Job respect**, shows how employees admire their peers and supervisors, e.g., *Mutual respect amongst the employees.*
- **Work projects**, companies projects, products and plans are included in this aspect, e.g., *Diverse products, numerous projects are provided by the company.*
- **Market viability**, provides information about the type and quality of the market, the company battles, e.g., *Competitive, changing market.*
- **Technology**, explores the technological aspects of the company, e.g., *The machinery used in this company is built on ancient technology.*
- **Work issues**, highlights the operational, legal and internal issues of the company, e.g., *Most of the machines are not working.*
- **Knowledge scope**, lists skills required by the company and knowledge acquired by the employees, e.g., *Technical knowledge in required for this task.*
- **Employee communication**, discusses the interaction between employees and the companies, e.g., *People are very interactive in this company.*
- **Stress**, stress and pressure the employees and companies feel, e.g., *Getting underpaid, stress good for working in a competitive environment.*

We also show the corpus frequency of these aspects in Table 2.

4.2 Assigning Polarity to the Aspects

In this section, we describe the process of assigning polarity to the aspects.

Universal Dependent Modifiers. Keeping in mind the goal of finding the polarity score of each aspect (out of the top 30 extracted aspects) present in a review, we start with finding the universal dependencies⁴ of aspects in the

⁴ <http://universaldependencies.org/>.

Table 2. Extracted aspects with their corpus frequency.

Aspect	Frequency	Aspect	Frequency
Employees/Co-workers	7659	Employee experience	1288
Work Life	7305	Location	627
Perks/Benefits	4565	Leadership	599
Office culture	4192	Technology	490
Working time	3658	Politics	451
Salary	3323	Flexibility	340
Management	2654	Company business	116
Job opportunities	2129		

review. We focus primarily on three dependencies, namely, *adjectival modifier* (*amod*), *adverbial modifier* (*advmod*) and *nominal subject* (*nsubj*).

For better understanding, we provide some examples from reviews in our dataset, with which we demonstrate the aforementioned dependencies associated with the aspects present.

- *Great opportunities for career growth.* *amod*(opportunities, Great)
- *Very political and conservative company. Old school, stodgy.* *advmod*(political, Very)
- *Great people to work with, perks of business traveling.* *nsubj*(travelling, perks)

We use *StanfordCoreNLP Parser* as the tool to extract these universal dependencies. After we find the dependencies, we use these ‘trigger’ words to determine the sentimental polarity of the corresponding aspect. As the aspect sentiment is determined by these modifiers, we lookup their polarities in the prepared sentiment dictionary. These polarities serve as the corresponding aspect score for that particular aspect. This process is repeated for the top 30 aspects that are found in the review.

Context Patterns. In the event that the trigger word is not in the sentiment dictionary, we move on to the second step in the ensemble. Here, we look at the context (window size - 5 words used, including the aspect). We try to find dependency patterns mentioned by [25] and use them to determine the overall polarity score for the aspect. Our assumption is that the presence of highly polar words in the context will contribute to the overall polarity of the aspect. Negations have been appropriately handled as they flip the polarity.

ELM Based Polarity Score. Should the two procedures mentioned above fail to assign a score to the aspect, we use the prediction made by our ELM model (1 - positive, 0 - negative). We directly lookup the polarity of the aspect word in the sentiment dictionary and adjust it based on the ELM output as per the following formula:

$$aspect_{score} = (e_{out}) * (lookup(aspect)) + (1 - e_{out}) * (-1 * lookup(aspect))$$

here, e_{out} is the output predicted by the ELM for the review, $lookup(aspect)$ is the polarity score of the aspect word as obtained from the sentiment dictionary.

ELM Based Semi Random Score. If the aspect word itself is not present in the sentiment dictionary, we initialize a random polarity value based on the ELM output. The following formula is used to generate the random score:

$$aspect_{score} = \begin{cases} rand(0, 1) & , e_{out} \equiv 1 \\ rand(-1, 0) & , e_{out} \equiv 0 \end{cases}$$

Here, $rand(a, b)$ is a random generator function which generates random real numbers within the range $[a, b]$.

We had to assign a score randomly to the aspects in only 2% of cases. This indicates that the semi-random polarity generation of the aspects did not impact the overall aspect-sentiment embeddings much. However, a fully automatic process is always desirable and as such, we plan on doing so in our future work.

5 Experimental Results

In this section, we describe the experimental results and insights drawn from the crawled datasets. Table 6 (Appendix A) shows the aspects and aspect terms that we extracted from the dataset.

We utilized both SVM and ELM models (Table 3) on the dataset in order to detect sentiment. In the experiments, the SVM method performed better than ELM in terms of accuracy. However, the difference between the performances of these classifiers on the given dataset was not statistically significant, as per the paired t-test ($p > 0.05$). Also, in the case of training time, we observed that the ELM method was almost 30 times faster than that of the SVM method on this dataset.

Table 3. Performance of SVM and ELM on the dataset.

Model	Accuracy	Macro F1-score
SVM	75.13%	74.85%
ELM	74.89%	75.09%

5.1 Aspect-Sentiment Embeddings

In Sect. 4.2, we described the process of assigning polarity to the aspects. We then calculated the score of an aspect belonging to a company by simply taking

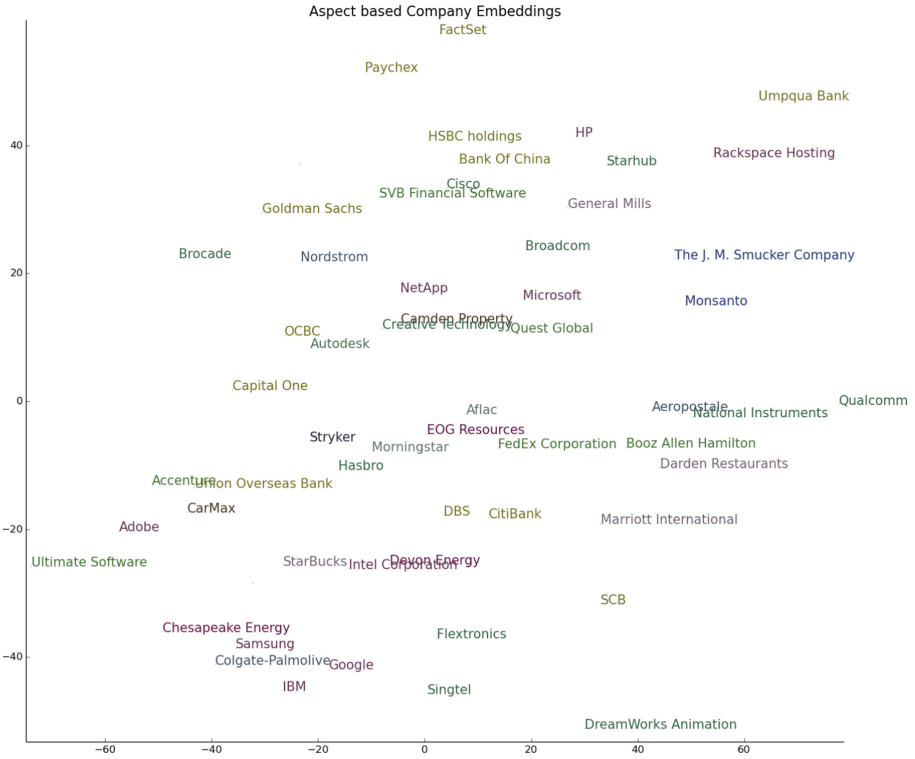


Fig. 2. Projection of the Aspect-sentiment Embeddings of the companies. Note: The same color represents companies from the same sector.

the average score of the polarities belonging to that aspect in all reviews of said company.

If we consider each aspect as a separate dimension, and the polarity value of a company for one aspect is the projection along that dimension, then we can project each company in a n -dimensional space where $n = \text{number of aspects}$. In our case, $n = 30$. In Fig. 2 we show projection of the companies using aspect-sentiment embeddings.

The motivation for constructing aspect-sentiment embeddings for the companies was to be able to calculate the similarities between companies based on the sentiments of the employees working at those companies.

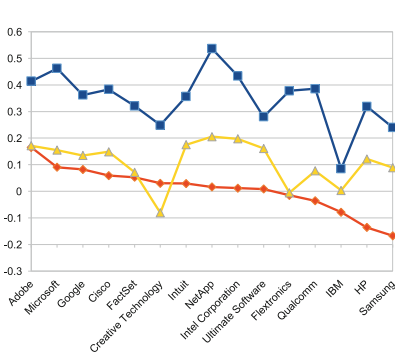
In Table 4, we present the cosine similarity scores between companies from similar or differing sectors. We see that even though *Goldman Sachs* and *DBS* are in same sector, i.e., Banking and Finance, they have a lower similarity score. However *DBS* and *SCB*(Standard Chartered Bank) have a relatively higher similarity score.

Table 4. Cosine similarities between the companies (calculated based on the aspect-sentiment embeddings)

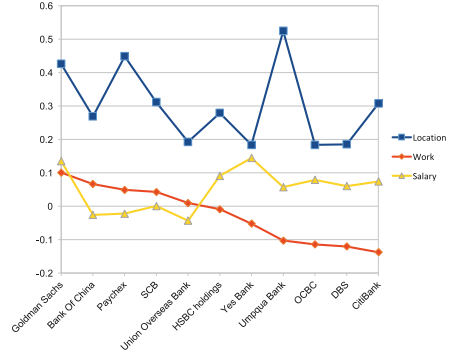
Company 1	Company 2	Cosine Similarity
Accenture	Booz Allen Hamilton	0.548
Accenture	FedEx	0.733
Google	Microsoft	0.549
Microsoft	Intel	0.486
Adobe	HP	0.370
Adobe	Google	0.276
Adobe	IBM	-0.214
OCBC	Goldman Sachs	0.422
Goldman Sachs	DBS	-0.098
DBS	SCB	0.313
Goldman Sachs	SCB	0.041
Singtel	Broadcom	0.424
Singtel	Starhub	-0.244
Microsoft	Stryker	0.687
National Instruments	Microsoft	-0.117
NetApp	Hasbro	0.655
Monsanto	Quest GLocal	-0.380

Table 5. Companies with best/worst salary and work culture rating in tech and finance sectors.

	Location		Salary		Work life	
	Tech	Finance	Tech	Finance	Tech	Finance
B	Microsoft	Umpqua Bank	Intel	Yes Bank	Adobe	Goldman S
E	Intel	Goldman S	Adobe	Goldman S	Microsoft	Bank of China
S	Adobe	SCB	Microsoft	HSBC	Google	SCB
T	Google	Citibank	Cisco	OCBC	Cisco	UOB
	HP	HSBC	Google	Citibank	FactSet	HSBC
W	IBM	Yes Bank	Creative	UOB	Samsung	Citibank
O	Creative	OCBC	Flextronics	Bank of China	HP	DBS
R	NetApp	DBS	FactSet	SCB	NetApp	OCBC
S	Cisco	UOB	Samsung	Umpqua Bank	Creative	Umpqua Bank
T	FactSet	Bank of China	HP	DBS	Intuit	Yes Bank



(a) Tech sector.



(b) Finance sector.

Fig. 3. The plot of the polarity of the aspects of companies in tech and finance sector.

Table 5 presents the best and worst companies in the technological and finance sectors based on sentiment values of the salary, location and work life aspects. Analysis (Fig. 3b and 3a) shows that employees are mostly happy with the salary they receive in both the finance and tech sectors. However, in relation to most banks, employees provide negative feedback on work culture. And, although tech companies receive positive feedback for their work culture, the intensity of such positivity is comparatively lower than salary satisfaction.

6 Conclusion

In this paper, we described the process of constructing aspect-sentiment embeddings of companies. In particular, we employed aspect-level sentiment analysis on the previously barely researched employee reviews of various companies available on the site Glassdoor. Several experimental insights of the data are given in this study. We addressed the overall employee sentiment on different aspect granularities, e.g., *salary*, *location*, *work life*, etc. This study presents a useful tool for companies to address employees' concerns and increase staff morale. On the other hand, job seekers will also be able to use this study to better find the best employers in their domain of interests.

Future work will mainly focus on considering the rating already given by the users in order to develop a user-product-sentiment model. A more comprehensive aspect-level sentiment analysis is also an important part of this future work.

Acknowledgment. This research was supported in part by the National Natural Science Foundation of China under Grant no. 61472266 and by the National University of Singapore (Suzhou) Research Institute, 377 Lin Quan Street, Suzhou Industrial Park, Jiang Su, People's Republic of China, 215123.

A Top Aspects and Their Aspect-Terms

Table 6. Top 30 aspects along with their respective aspect terms.

Aspects	Aspect terms	Aspects	Aspect terms
Company business	Professional, booming, structured challenging, competitive, steady	Career development	Rewarding, international, guided challenging, difficult, solid
Employee communication	Strong, transparent, cryptic, remote, awful, effective	Office culture	Cooperative, balanced, exciting, suffered, abysmal, bias
Employees/co-workers	Excellent, cooperative, competent, stagnant, unfriendly, pretend	Employee experience	Diverse, useful, firsthand, horrific, odd, international
Flexibility	Strict, dependent, tremendous, encourage, minimal, great	Personal growth	Exponential, poor, constant, hierarchy, potential, constrain
Work issues	Legal, serious, inherent, internal, operational, demographic	Overall job	Excellent, temporary, overnight, changing, tough, secure
Knowledge scope	Immense, required, sharing, technical, limited, vast	Leadership	Appreciate, strong, poor, unwilling, dedicated, driving
Location	Strategic, remote, accessible, attractive, multiple, uncertain	Management	Flexible, fluctuate, inexperienced, mindful, dishonest, focus
Market viability	Changing, shrinking, impact, competitive, unknown, successful	Job opportunities	Excellent, driven, mindset, international, lacking, unique
Perks/Benefits	Unique, scares, incredible, illusion, lousy, incentives	Performance	Personal, necessary, measurable, extraordinary, encouraged, technical
Politics	Dysfunctional, drive, dirty, extreme, everywhere, internal	Work projects	Numerous, diverse, challenging, pushed, creative, unbearable
Job respect	Professional, mutual, utmost, solid, diminishing, great	Salary	Optimal, advancement, hikes, midrange, fantastic, unattractive
Official Staff	Understanding, excellent, competent, mean, motivated, dysfunctional	Stress	Underpaid, excessive, good, constant, additional, incompetent
Company support	Excellent, tedious, benefits, supervisors, on site, rare	Technology	Excellent, ancient, global, latest, green, innovative
Working time	Exciting, peak, tough, stressful, extra, irregular	Job training	Prepares, competent, notch, outstanding, outdated, tough
Vacation	Decent, mandatory, paid, planned, considering, balance	Work life	Competent, mundane, exciting, friendly, versatile, stressful

References

1. Baccianella, S., Esuli, A., Sebastiani, F.: Sentiwordnet 3.0: an enhanced lexical resource for sentiment analysis and opinion mining. In: LREC 2010, vol. 10, pp. 2200–2204 (2010)
2. Blitzer, J., Dredze, M., Pereira, F., et al.: Biographies, bollywood, boom-boxes and blenders: domain adaptation for sentiment classification. In: ACL 2007, vol. 7, pp. 440–447 (2007)
3. Bollegala, D., Weir, D., Carroll, J.: Cross-domain sentiment classification using a sentiment sensitive thesaurus. *IEEE Trans. Knowl. Data Eng.* **25**(8), 1719–1731 (2013)
4. Cambria, E., Huang, G.-B., et al.: Extreme learning machines. *IEEE Intell. Syst.* **28**(6), 30–59 (2013)
5. Cambria, E., Poria, S., Hazarika, D., Kwok, K.: SenticNet 5: discovering conceptual primitives for sentiment analysis by means of context embeddings. In: AAAI, pp. 1795–1802 (2018)
6. Cambria, E., Schuller, B., Xia, Y., White, B.: New avenues in knowledge bases for natural language processing. *Knowl.-Based Syst.* **108**, 1–4 (2016)
7. Cambria, E., Song, Y., Wang, H., Howard, N.: Semantic multi-dimensional scaling for open-domain sentiment analysis. *IEEE Intell. Syst.* **29**(2), 44–51 (2014)
8. Chaturvedi, I., Ong, Y.-S., Tsang, I., Welsch, R., Cambria, E.: Learning word dependencies in text by means of a deep recurrent belief network. *Knowl.-Based Syst.* **108**, 144–154 (2016)
9. Denecke, K.: Using sentiwordnet for multilingual sentiment analysis. In: Data Engineering Workshop at ICDEW 2008, pp. 507–512. IEEE (2008)
10. Ding, X., Liu, B., Yu, P.S.: A holistic lexicon-based approach to opinion mining. In: 2008 International Conference on Web Search and Data Mining, pp. 231–240. ACM (2008)
11. Esuli, A., Sebastiani, F.: Sentiwordnet: a publicly available lexical resource for opinion mining. In: Proceedings of LREC, vol. 6, pp. 417–422 (2006)
12. Hu, M., Liu, B.: Mining and summarizing customer reviews. In: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 168–177. ACM (2004)
13. Xia, H., Tang, J., Gao, H., Liu, H.: Unsupervised sentiment analysis with emotional signals. In: WWW 2013, pp. 607–618 (2013)
14. Huang, G.-B.: An insight into extreme learning machines: random neurons, random features and kernels. *Cogn. Comput.* **6**(3), 376–390 (2014). <https://doi.org/10.1007/s12559-014-9255-2>
15. Huang, G.-B., Chen, L., Siew, C.-K.: Universal approximation using incremental constructive feedforward networks with random hidden nodes. *IEEE Trans. Neural Netw.* **17**(4), 879–892 (2006)
16. Huang, G.-B., Wang, D.H., Lan, Y.: Extreme learning machines: a survey. *Int. J. Mach. Learn. Cybern.* **2**(2), 107–122 (2011)
17. Huang, G.-B., Zhou, H., Ding, X., Zhang, R.: Extreme learning machine for regression and multiclass classification. *IEEE Trans. Syst. Man Cybern. Part B: Cybern.* **42**(2), 513–529 (2012)
18. Le, Q.V., Mikolov, T.: Distributed representations of sentences and documents. In: ICML, vol. 14, pp. 1188–1196 (2014)
19. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint [arXiv:1301.3781](https://arxiv.org/abs/1301.3781) (2013)

20. Miller, G.A.: Wordnet: a lexical database for English. *Commun. ACM* **38**(11), 39–41 (1995)
21. Moniz, A., de Jong, F.: Sentiment analysis and the impact of employee satisfaction on firm earnings. In: de Rijke, M., et al. (eds.) *ECIR 2014. LNCS*, vol. 8416, pp. 519–527. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-06028-6_51
22. Ohana, B., Tierney, B.: Sentiment classification of reviews using sentiwordnet. In: 9th. *IT and T Conference*, p. 13 (2009)
23. Pan, S.J., Ni, X., Sun, J.-T., Yang, Q., Chen, Z.: Cross-domain sentiment classification via spectral feature alignment. In: *WWW 2010*, pp. 751–760. ACM (2010)
24. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up?: Sentiment classification using machine learning techniques. In: *EMNLP 2002*, vol. 10, pp. 79–86. ACL (2002)
25. Poria, S., Agarwal, B., Gelbukh, A., Hussain, A., Howard, N.: Dependency-based semantic parsing for concept-level text analysis. In: Gelbukh, A. (ed.) *CICLing 2014. LNCS*, vol. 8403, pp. 113–127. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-54906-9_10
26. Poria, S., Cambria, E., Gelbukh, A.: Aspect extraction for opinion mining with a deep convolutional neural network. *Knowl.-Based Syst.* (2016)
27. Ridella, S., Rovetta, S., Zunino, R.: Circular backpropagation networks for classification. *IEEE Trans. Neural Netw.* **8**(1), 84–97 (1997)
28. Shah, R.R.: Multimodal analysis of user-generated content in support of social media applications. In: *Proceedings of the International Conference on Multimedia Retrieval (ICMR)*, pp. 423–426. ACM (2016)
29. Shah, R.R., Yu, Y., Verma, A., Tang, S., Shaikh, A.D., Zimmermann, R.: Leveraging multimodal information for event summarization and concept-level sentiment analysis. In: *Proceedings of the Knowledge-Based Systems (KBS)*, pp. 1–8 (2016)
30. Shah, R.R., Yu, Y., Zimmermann, R.: Advisor: personalized video soundtrack recommendation by late fusion with heuristic rankings. In: *Proceedings of the International Conference on Multimedia (MM)*, pp. 607–616. ACM (2014)
31. Turney, P.D.: Thumbs up or thumbs down?: Semantic orientation applied to unsupervised classification of reviews. In: *ACL 2002*, pp. 417–424. ACL (2002)
32. Vogl, T.P., Mangis, J.K., Rigler, A.K., Zink, W.T., Alkon, D.L.: Accelerating the convergence of the back-propagation method. *Biol. Cybern.* **59**(4–5), 257–263 (1988)



A Decision-Level Approach to Multimodal Sentiment Analysis

Haithem Afli¹(✉), Jason Burns², and Andy Way²

¹ ADAPT Centre, Department of Computer Science, Munster Technological University, Cork, Ireland

haithem.afli@adaptcentre.ie

² ADAPT Centre, School of Computing, Dublin City University, Dublin, Ireland
andy.way@adaptcentre.ie

Abstract. There has been near exponential increase in the use of images and video on various Social Media platforms in the last few years, in place of or in addition to the use of plain text. Automated sentiment analysis, at its core, is the capturing of human emotion by machine - the addition of image and video to social media output had made this already challenging task even greater. In this paper, we propose a multimodal, decision-level based approach to sentiment analysis (SA) of Twitter feeds. The solution proposed and outlined in this paper, combines the sentiment analysis scoring of not just text-based output but integrates SA scoring generated from analysis of image captions. For our experiments, we focused on politics and on two political topics (Trump/Brexit) that are generating a lot of discussion and debate on Twitter. We chose the political domain given the power that Social Media has on possibly influencing voters (<https://www.theguardian.com/technology/2016/jul/31/trash-talk-how-twitter-is-shaping-the-new-politics>) and the ‘strong’ opinions that are expressed in this area.

Keywords: Multimodality · Sentiment analysis · Image captioning · Tweets

1 Introduction

The purpose of combining multi-modal data in order to produce better estimates for sentiment analysis, in other words improving the accuracy of estimating human opinion when expressed through a combination of multi-modal channels, is the goal of this work. To date, there has been limited research into multimodal sentimentality [3]. The growth in social media has been extraordinary and the power of certain social media platforms at the very least to disseminate information globally in a few minutes of the event happening is bound to have an effect on a large proportion of people’s opinion. So, given the global reach and effect of social media, the ability to accurately monitor and analyse information, and the sentiment scoring of that information, is crucial for a number of reasons. But the

impact of media on world events is nothing new and as far back as 1971 research was being conducted on the influence of the media and world events on the stock market [10]. It is the informal structure (and use of ‘non-canonical language’ [5]) and content of social media output, and its sheer volume, that has led to several challenges in accurately processing and obtaining accurate sentiment analysis (SA) scores. Certain social media channels (such as Twitter) present their own challenges with either a limited character count, informal language usage, a lack of context and now an ever-increasing use of images and video.

These challenges and need for obtaining accurate SA scores has led to a large amount of successful research on sentiment analysis of social media, largely focusing on some derivative of a text classification process [2,9]. The analysis, for most part, makes a distinction between positive sentiment, negative sentiment, or neutral. But it is the ever-increasing use of multi-modal communication, namely images and video in addition to or as a replacement for text, that has led to a major challenge for any trying to conduct accurate SA on social media output. Up to recently, there is not a lot of work being carried out in addressing this ‘multi-modal’ sentiment problem where several different methods of communication are being used [11].

Twitter is one of the more common social media platforms used by people to express their opinions and emotions. The access available to harness and process tweets (for example, through the use of APIs) has led to some interesting studies from determining how certain users of Twitter can affect general opinion on the platform through to establishing the power of twitter on the outcome of US elections [12].

What is clear is that given the character restriction of Twitter to 140 characters and the power of images to convey a complex amount of information very quickly, there is a very large increase in the amount of images being included in tweets. What is also being established is that tweets are likely to be far more influential if they do contain an image. In 2012 for example, arguably the most popular tweet of the year was a simple image of Barack and Michelle Obama with little or no text included in many of the retweets.

However it is in the combining of information from multiple modes that is essential in improving the accuracy of sentiment analysis applications. Humans communicate in a multi-modal formats and therefore the analysis of text, audio and images/video is fundamental to improving our sentiment analysis accuracy levels [9]. The reason for this increased interest in multi-modal sentiment analysis is directly linked to the massive increase in the use of images and video through social media (Facebook and Youtube) in particular.

There are many practical applications of automated sentiment analysis in use today [7]. To just look at one example, as more and more product reviews are conducted through video, the market is more interested in learning opinions expressed through this medium than simply based on textual reviews. Therefore, marketing teams worldwide are looking at solutions where sentiment is more accurately tracked in relation to the use of their particular product or service. But the automation of SA is challenging when limiting your analysis to text

alone. As can be seen in the Fig. 2, the text is neutral in terms of sentiment, but it is clear from the images that the sentiment is positive simply by the smiling faces that are included with the tweet. This is a straight forward and simple, but powerful example of how images can enrich the sentiment that we can receive from tweets (Fig. 1).



Fig. 1. Sample tweet of positive sentiment

For this work, we focused on Twitter as the social media channel of choice. The use of image and video, combined with text, is growing rapidly in Twitter and for good reason. A study by Buffer showed that including a simple image with your tweet was the most advantageous method of ensuring your tweet was more widely read. Studies have recently been carried out on the effect of adding multimedia to tweets within Sian Weibo (Chinese version of Twitter) and it is clear that adding media (particularly images) does increase the popularity and lifetime of that tweet [13].

The goal is to improve on the existing textual sentiment analysis solutions by including SA scores generated from image captions in order to determine if the overall SA score is improved by using this additional information. By using the very latest in image caption generation technology, we can unlock the information contained within the embedded image to give us a more accurate score on what sentiment is being expressed by a particular tweet.

2 Related Work

Based on the advances in image recognition, attention is now starting to focus on sentiment analysis of images and video, and this is too proving to be a major challenge. It is one thing to recognise objects in an image, possibly connect those objects and offer a context, but it is a major leap in understanding to try to determine an opinion or emotion from the image. The research is hampered by the lack of available datasets that properly annotate images with emotion tags, a suitable lexicon of images and associated sentiment categories. A few public datasets such as the International Affective Picture System (IAPS)¹ and the Geneva Affective Picture Database (GAPED) [4] being the only datasets available of any note that provide either ratings or annotations for emotion and sentiment.

The first step in sentiment analysis of images is to determine the content of images and form a sentence describing the content - it is substantially harder than just classifying images [8]. The human eye can gather an immense amount of information from glancing at an image, a capability that is proving to be very difficult for machines to replicate [6]. There are several reasons why conducting object recognition/image captioning on images and video is very difficult; not least because a very large dataset is needed to train models - in many instances a model needs a large amount of prior knowledge to compensate for the information that is missing [8].

3 Decision-Level Approach to Multimodal Sentiment Analysis

There were several phases to the project that are outlined in the following sections. In summary, searches were conducted on Twitter for specific popular topics (#Trump, #Brexit) and only tweets with images were stored and processed on IBM Bluemix before being analysed for sentiment using the SentiTweetWords [1] analysis tool and NeuralTalk2².

3.1 Data Preparation and System Architecture

We chose the IBM Bluemix Platform as a Service (PaaS) environment to collect and process the tweets, a cloud based service making it easier to access the technology needed to process the tweets received. First we created a nodered.js app that would connect the Twitter API to our couchdb database (Cloudant) . Connected to the Twitter Decahose we searched for the hastags #Trump and #Brexit, and then stored the captured tweets in the Cloudant no-SQL database (based on the couchDB framework) as separate JSON objects. The JSON files were then copied, using Spark-based ETL tool, to the DashDB datawarehouse.

¹ <http://csea.php.ufl.edu/media/iapsmessage.html>.

² <https://github.com/karpathy/neuraltalk2>.

Once the JSON objects were copied to DashDB it was easier to perform basic analysis to isolate the tweets with images included - the nominated tweets (and images) were then exported from Bluemix for processing.

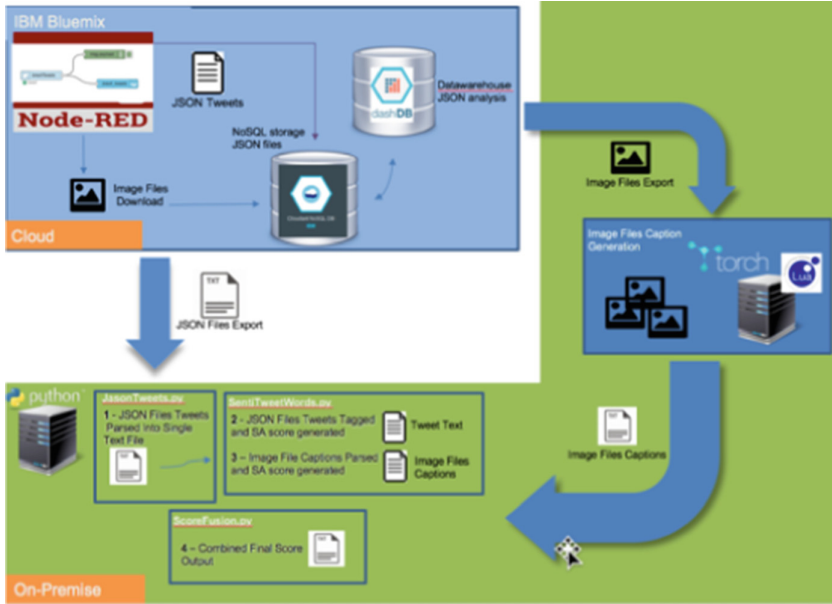


Fig. 2. Sample Tweet of positive sentiment

4 Experiments and Results

The collated scores for the tweet text and image captions were captured and averaged for analysis. In order to fully assess the sentiment analysis scoring for both text (SA_Text_Score) and image captions (SA_Image_Score) a longer scope of manual annotation would need to be undertaken -this is beyond the scope of this paper. However, taking a sample of 200 tweets, we were able to perform some elementary analysis of the scoring returned by the automated multimodal sentiment analysis application. Before addressing the results and scoring, there are some important with regard to our application and the general sentiment analysis environment:

- Image Recognition Maturity - In line with many leading-edge sub domains within the overall Artificial Intelligence/Machine Learning field, there is still a lot of scope for progression in the accuracy and completeness of image caption generation. Captions generated by NeuralTalk2 to are largely accurate but simple in content, therefore this affected the overall SA_Image_Score result.

- Informal Language Usage - Achieving an accurate SA score on tweets is difficult when sarcasm and very informal language is being used. This in turn affects and NLP application that is trying to determine sentiment, and therefore SA_Text_Score.
- Domain Choice - our domain of choice was Politics, and two popular topics in that area as of 2017: Brexit and Trump. As already discussed earlier in this document, we chose this domain because of the affect social media platforms can have on popular political opinion. However, it created some unique difficulties. For example, many prolific tweeters (and regular political commentators) expect that their political views are already known, and therefore provide little context to their tweets which makes it difficult to automate the SA process. Also, images can be used that are in direct conflict (in terms of sentiment) to the text of the tweet (sarcasm being used) and this can have an adverse affect on the overall averaged sentiment analysis score.

Both SA_Text_Score and SA_Image_Score had a sentiment analysis scoring range of between 0 to 1. Looking more closely at the actual results, focusing on the random 200 tweets, we observed the following:

- Close to 60% of tweets had SA_Text_Score and SA_Image_Score numbers that were very close in score (difference of $<.19$). There are several different interpretations to be taken from this - where the application can not determine a score it will default to neutral. It is therefore reassuring that both image and text correspond.
- $<9\%$ of scores were in contradiction to each other in terms of sentiment scoring - this is not necessarily a bad result. There are several possible reasons for this, some tweets contain a lot of textual information (relating to several different points) and therefore it is difficult to determine the sentiment accurately regardless of the image. As mentioned before, the use of sarcasm can confuse the application, therefore this is something that could be used as a trigger for detecting sarcasm
- When the image caption was accurate and the text of the tweet was clear to read, there was a very close correlation between SA_Text_Score and SA_Image_Score. $\bar{1}0\%$. Given the challenges that are in place for our application this again was a positive result highlighting the fact that when some obstacles were removed, the application was accurate in predicting an SA score.

As can be seen in the example of Fig. 3, an image can either confirm opinion or present an alternative opinion, especially where the use of sarcasm is being employed.



Fig. 3. Sample Tweet of positive sentiment

5 Conclusion and Future Work

We have presented a multimodal sentiment analysis that builds on the existing work done on textual sentiment analysis and image captioning. By combining the different modes, we have shown that you can enrich the sentiment analysis score of the tweet by including the image caption in the sentiment scoring. There are some difficult challenges that need to be overcome, but we have presented some possible improvements that might go some way to alleviating the factors that affect accurate automated SA scoring. Twitter is a fast-moving and prolific medium for opinion expression, and focusing on the political domain served to highlight clearly the range, diversity and polarity of opinions that are expressed and shared on the Twitter platform. What is clear from our research, is that images and video are only going to increase in usage and it is no longer viable for a social media sentiment analysis application to ignore those media.

Acknowledgements. This work was supported by ADAPT (www.adaptcentre.ie), which is funded under the SFI Research Centres Programme (Grant 13/RC/2016) and is co-funded by the European Regional Development Fund.

References

1. Afli, H., McGuire, S., Way, A.: Sentiment translation for low resourced languages: experiments on Irish general election tweets. In: Proceedings of the 18th International Conference on Computational Linguistics and Intelligent Text Processing, Budapest, Hungary (2017)
2. Baecchi, C., Uricchio, T., Bertini, M., Del Bimbo, A.: A multimodal feature learning approach for sentiment analysis of social network multimedia. *Multimedia Tools Appl.* **75**(5), 2507–2525 (2016)
3. Cambria, E., Schuller, B., Xia, Y., Havasi, C.: New avenues in opinion mining and sentiment analysis. *IEEE Intell. Syst.* **28**(2), 15–21 (2013)
4. Dan-Glauser, E.S., Scherer, K.R.: The geneva affective picture database (GAPED): a new 730-picture database focusing on valence and normative significance. *Behav. Res. Methods* **43**(2), 468 (2011)
5. Eisenstein, J.: What to do about bad language on the internet. In: Proceedings of NAACL-HLT 2013 Atlanta, Georgia, pp. 359–369 (2013)
6. Karpathy, A., Fei-Fei, L.: Deep visual-semantic alignments for generating image descriptions. *IEEE Trans. Pattern Anal. Mach. Intell.* **39**(4), 664–676 (2017)
7. Kazemian, S., Zhao, S., Penn, G.: Evaluating sentiment analysis evaluation: a case study in securities trading. In: Proceedings of the 5th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, Baltimore, Maryland, USA, pp. 119–127 (2014)
8. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. *Commun. ACM* **60**(6), 84–90 (2017)
9. Morency, L.-P., Mihalcea, R., Doshi, P.: Towards multimodal sentiment analysis: harvesting opinions from the web. In: Proceedings of the 13th International Conference on Multimodal Interfaces, ICMI '11, New York, NY, USA, pp. 169–176. ACM (2011)
10. Schumaker, R.P.: An analysis of verbs in financial news articles and their impact on stock price. In: Proceedings of the NAACL HLT 2010 Workshop on Computational Linguistics in a World of Social Media, WSA Stroudsburg, PA, USA, vol. 10, pp. 3–4. Association for Computational Linguistics (2010)
11. Shah, R.R.: Multimodal-based multimedia analysis, retrieval, and services in support of social media applications. In: Proceedings of the 2016 ACM on Multimedia Conference, MM 2016, New York, NY, USA, pp. 1425–1429. ACM (2016)
12. Wang, H., Can, D., Kazemzadeh, A., Bar, F., Narayanan, S.: A system for real-time twitter sentiment analysis of 2012 U.S. presidential election cycle. In: Proceedings of the ACL 2012 System Demonstrations, ACL 2012, Stroudsburg, PA, USA, pp. 115–120. Association for Computational Linguistics (2012)
13. Zhao, X., Zhu, F., Qian, W., Zhou, A.: Impact of multimedia in sina weibo: popularity and life span. In: Semantic Web and Web Science - 6th Chinese Semantic Web Symposium and 1st Chinese Web Science Conference, CSWS 2012, Shenzhen, China, 28–30 November, pp. 55–65 (2012)



Text-Image Sentiment Analysis

Qian Chen¹, Edoardo Ragusa², Iti Chaturvedi¹, Erik Cambria¹(✉),
and Rodolfo Zunino²

¹ School of Computer Science and Engineering, Nanyang Technological University,
Nanyang 50 Ave, Singapore, Singapore

CAMBRIA@ntu.edu.sg

² Department of Electrical, Electronic, Telecommunications Engineering and Naval
Architecture, DITEN, University of Genoa, Genova, Italy

Abstract. Expressiveness varies from one person to another. Most images posted on Twitter lack good labels and the accompanying tweets have a lot of noise. Hence, in this paper we identify the contents and sentiments in images through the fusion of both image and text features. We leverage on the fact that AlexNet is a pre-trained model with great performance in image classification and the corresponding set of images are extracted from the web. In particular, we present a novel method to extract features from Twitter images and the corresponding labels or tweets using deep convolutional neural networks trained on Twitter data. We consider fine tuning AlexNet pre-trained CNNs to initialize the model and AffectiveSpace of English concepts as text features. Lastly, to combine the image and text predictions we propose a novel sentiment score. Our model is evaluated on Twitter dataset of images and corresponding labels and tweets. We show that accuracy by merging scores from text and image models is higher than using any one system alone.

Keywords: Sentiment analysis · Text-image joint · Weighted score

1 Introduction

The proliferation of Web 2.0 technologies and the increasing use of computer-mediated communication resulted in exponential growth of information online. Despite the Internet's role as a facilitator of information in this Big Data Era, it has overloaded users with unrelated and noisy data. It is urgent to find an efficient and high-performance approach which extracts useful features from this massive amount of data. Natural Language Processing (NLP), a subdomain of Artificial Intelligence (AI), comes as a useful and practical method to handle the analysis of the language that humans use naturally in order to connect with computers and machines in both written and spoken contexts. For more than three decades, NLP has been handling problems between human and computer interaction. NLP major tasks involve named entity recognition (NER), sentiment analysis, speech recognition, information retrieval, information extraction, relationship extraction, parsing, and machine translation, among others [8].

© Springer Nature Switzerland AG 2023

A. Gelbukh (Ed.): CICLing 2018, LNCS 13397, pp. 169–180, 2023.

https://doi.org/10.1007/978-3-031-23804-8_14

Sentiment analysis, one of its most interesting and challenging tasks, combines advanced techniques from NLP, machine learning, and information retrieval to extract opinions and subjective knowledge from online messages in social media. In fact, the rise and expansion of social media enabled millions of users to share their views, lives and interests in an impromptu manner and in real time, creating a huge amount of sentiment lexicons to be extracted and analyzed [7, 9, 15]. Initially, sentiment analysis focused on text documents such as product reviews and comments posted on social media platforms (e.g., Facebook, Twitter and Weibo). From the text, it was possible to extract the sentiment polarities of the sentences and classify them into positive, neutral and negative. It has been proposed that sentiment classifiers for text can be trained from positive and unlabeled examples using machine learning techniques [19] such as Naïve Bayesian method and Support Vector Machines (SVM). Afterwards, sentiment analysis methods based on sentiment lexicons appeared and neural network enabled considerable progress in text sentiment classification.



(a) Positive



(b) Negative

Fig. 1. Examples for image polarities

Since the popularization of multimedia content in various social networks, text is no longer the only mode for sharing information. Image and videos are enabling people to express their thoughts and sentiments easily [12]. As a consequence in this Big Data Era [10], sentiment analysis cannot be limited to text domain. In particular, images play a more important role in sentiment analysis, since they have the fullest quality of information [14]. Furthermore, videos can also be represented as a sequence of images. For example, YouTube videos are a convenient way to share news events and product descriptions. Figure 1 (a) illustrates an image of two gentlemen paddling their canoes and laughing, annotated as positive polarity; and (b) illustrates an image of a building in ruins annotated as negative polarity.

CNN serves as the mainstream technique for image processing which can distinguish classes of images properly. Several authors have used CNNs for object recognition and classification of images [13, 18, 30, 31]. It has been proved that Deep Convolutional networks were ideal for image sentiment analysis as it was

able to detect over 10,000 different objects simultaneously. This powerful CNN architecture named AlexNet [18] is used in our method. Complementary to this, there are abundant Flickr datasets containing more than 3,000 Adjective Noun Pairs (ANPs), each image belongs to one ANP and the number of images in one ANP is ranging from dozens to thousands [13]. On the basis of ANPs, different levels of different emotions may be extracted [4, 30].

In this paper, we propose a method using textual and visual features to predict the sentiment polarity of Tweets containing both image and text. Following is the structure of this paper: Sect. 2 introduces studies related to sentiment analysis and the state-of-the-art methods; Sect. 3 is the preliminaries for our approach; Sect. 4 states our methods in a detailed way; Sect. 5 is the results and evaluation for our methods; finally, we summarize our work in Sect. 6.

2 Related Work

Traditional methods for sentiment analysis are mainly applied to text mining, which do not consider the presence of multimodal data, e.g., videos or images [24, 31]. As one of popular data format, images present more information but are more complex in compare to text.

As a novel and widely applied branch of NLP, sentiment analysis is to analyze the sentiment polarity of data, namely the attitudes, emotions and opinions of the data, which can be applied in marketing decision and product optimization for companies, as well as purchase decision for individual customers. A wheel of emotions created by Plutchik sorts emotions into 8 primary bipolar emotions and makes emotions in a colorful wheel showing the connection between emotions and colors [23]. Plutchik's Wheel of Emotions provides the basis for sentiment analysis, making it easier to handle nuanced semantic concepts and intuitively presenting sentiments in term of visual perception. Cambria et al. proposed the hourglass model inspired by Plutchik's studies [28]. Roughly, we divide sentiments into three polarities: positive, neutral and negative. In the 3D hourglass model, affective states from strongly positive to null to strongly negative are shaped to an hourglass containing four basic emotions: Attention, Aptitude, Pleasantness and Sensitivity. It uses basic emotions pairwise to result complex emotions.

ConceptNet [20] is a representation of the Open Mind Common Sense corpus representing noun phrases, verb phrases, adjective phrases or prepositional phrases by concept nodes. WordNet-Affect is a linguistic resource for the lexical representation of affective knowledge containing 1,903 terms referring to mental [27]. By applying the blending technique on ConceptNet and WordNet-Affect, Cambria et al. developed AffectiveSpace, a suitable knowledge base for emotive reasoning [6]. AffectiveSpace contains 100-dimensional concept embeddings, which can be embedded in potentially any cognitive system dealing with real-world semantics.

In 2012, Siersdorfer et al. predicted sentiment of images using color histograms and Scale-Invariant Feature Transform (SIFT) techniques dataset with

more than half a million Flickr images [26]. They used SentiWordNet as query terms to gather images with sentiment orientations. The bag-of-visual words representation and the color distribution of images are used to learn the image features. Through studying the connection between sentiment of images expressed in metadata and their visual content, Siersdorfer et al. achieved the precision values of up to 70% but with low recall values. Zhang et al. processed Sentiment Analysis on Microblogging by integrating text and image features [32]. In 2016, Katsurai et al. proposed a method mapping visual, textual and sentiment views into the latent embedding space and using correlations among these features [17]. The visual features is learnt from color histogram of images and this method achieved an accuracy of 74.77% on Flickr dataset and 73.60% on Instagram dataset.

DeepSentiBank [13] containing more than 3,000 ANPs significantly improved in both annotation accuracy and retrieval performance [22], compared to its predecessors which mainly use binary SVM classification models. Based on neural networks, CNNs introduced convolutional filters to extract features and obtained outstanding achievements in image processing and deep learning. You et al. proposed a progressive CNN architecture [31] on CAFFE [16]. They trained half a million samples with ANPs from Flickr and fine-tuned the deep network using a progressive strategy therefore obtained a considerable accuracy with high recall. You et al. proposed to use CNNs for the extraction of visual features and made fusions with textual features extracted from an unsupervised language model by learning distributed representations for documents and paragraphs [30]. Their model achieved a precision of 0.776 with recall of 0.740 by Early Fusion. Campos et al. provided a deep-dive analysis into *CaffeNet* and presented several experiments studying for the task of visual sentiment prediction [11].

3 Preliminaries

In this section, we will give the theoretical basis about CNNs and AffectiveSpace 2 for our method.

3.1 Deep Convolutional Neural Network

CNNs are a specific class of neural networks, based on four main building blocks: convolutions (kernels), non-linearities, pooling and dropout layers. A CNN is comprised of one or more convolutional layers (kernels) alternated by non linearities. Between them are inserted pooling and dropout layers. Finally, one or more fully connected layers as in a standard neural network gives the classification results.

Each convolutional layer works as a feature extractor. Using objects recognition as an example, lower level detects simple features like straight edges, simple colors, and curves, higher level extracts more complex features like noses, eyes. Typical non linearities are *ReLU* and *Tanh*. Dropout layer are a regularization

technique for reducing overfitting in neural networks by preventing complex co-adaptation on training data. Max pooling layers performs down-sampling by dividing the input into pooling regions, and computing the maximum of each region. The fully connected layer merges the extracted feature in order to perform the classification task.

This models present a huge number of parameters and are trained using standard back propagation techniques. Training from scratch requires labeled datasets with millions of patterns. For this reason in many applications transfer learning is applied. Transfer learning consist in remove the last fully-connected layer from a fully trained CNN, and replacing it with a new one. Then fine-tuning is applied to the weights of the new network by continuing the back -propagation. The main advantage of this technique is that it is possible to exploit feature detector for similar tasks, as an example adapt features for object recognition to polarity detection.

3.2 AffectiveSpace 2

To improve our model with textual features, we projected textual tweets data to AffectiveSpace 2. It is an effective way to cope with the evergrowing number of concepts and semantic features using AffectiveSpace. Cambria et al. replaced singular value decomposition (SVD), a low-rank approximation method, with random projection (RP) [3] to map the original high-dimensional data-set into a much lower-dimensional subspace by using a Gaussian $N(0, 1)$ matrix, while preserving the pair-wise distances with high probability. This follows Johnson and Lindenstrauss's (JL) Lemma [2]. The JL Lemma states that with high probability, for all pairs of points $x, y \in X$ simultaneously, there is:

$$\sqrt{\frac{m}{d}} \|x - y\|_2 (1 - \varepsilon) \leq \|\Phi_x - \Phi_y\|_2 \leq \sqrt{\frac{m}{d}} \|x - y\|_2 (1 + \varepsilon) \quad (1)$$

where X is a set of vectors in Euclidean space, d is the original dimension of this Euclidean space, m is the dimension of the space we wish to reduce the data points to, ε is a tolerance parameter measuring to what extent is the maximum allowed distortion rate of the metric space, and Φ is a random matrix.

Sarlos introduced that structured random projection for making matrix multiplication much faster [25]. Achlioptas and Dimitris proposed sparse random projection [1] to replace the Gaussian matrix with i.i.d. entries in:

$$\phi_{ji} = \sqrt{s} \begin{cases} 1 & \text{with prob. } \frac{1}{2s} \\ 0 & \text{with prob. } 1 - \frac{1}{s} \\ -1 & \text{with prob. } \frac{1}{2s} \end{cases} \quad (2)$$

where one can achieve a $\times 3$ speedup by setting $s = 3$, since only one third of the data need to be processed.

When the number of features is much larger than the number of training samples ($d \gg n$), subsampled randomized Hadamard transform (SRHT) is preferred, as it behaves very much like Gaussian random matrices but accelerates the process from $O(nd)$ to $O(n \log d)$ time [21]. From [21, 29], for $d = 2^p$ where p is any positive integer, a SRHT can be defined as:

$$\phi = \sqrt{\frac{d}{m}} R H D \quad (3)$$

where m is the number we want to subsample from d features randomly; R is a random $m \times d$ matrix (the rows of R are m uniform samples from the standard basis of \mathbb{R}^d); $H \in \mathbb{R}^{d \times d}$ is a normalized Walsh-Hadamard matrix, which is defined recursively: $H_d = \begin{bmatrix} H_{k/2} & H_{k/2} \\ H_{k/2} & H_{k/2} \end{bmatrix}$ with $H_2 = \begin{bmatrix} +1 & +1 \\ +1 & 1 \end{bmatrix}$ and D is a $d \times d$ diagonal matrix and the diagonal elements are i.i.d. Rademacher random variables.

AffectiveSpace 2 is a vector space model preserving the semantic and affective relatedness of common-sense concepts while being highly scalable. In our method, it is an important part to extract sentiment features from textual Twitter data using AffectiveSpace 2.

4 Proposed Framework

This paper proposed approach consists in merging the information from images and their captions. The feature from the image and the caption are extracted independently. The text features are extracted by means of single world projections on affectspace 2. The visual feature are extracted using AlexNet and fine-tuned by Twitter images.

4.1 Visual Features

The proposed feature extraction model of the CNN is inspired from AlexNet. Since AlexNet is a deep CNN architecture trained on 1.2 million images for the task of object detection with considerable precisions, it is efficient to detect sentiment of images by fine-tuning AlexNet with labeled images. In our model, we removed the fully connected layers and replaced it with a fully connected layer of size 4096×2 . Then we fine tune the weights using about 18928 pattern from Twitter, derived from 301 negative images and 581 positive images.

4.2 Textual Features

The textual features are extracted by 5-fold cross-validation method with AffectiveSpace. The original samples are randomly partitioned into 5 equal sized subsamples. Of the 5 subsamples, we train four of them and the other subsample is retained as the validation data. This process is repeated 5 times. Supporting Vector Machines (SVM) is used in the procedure of extracting textual features.

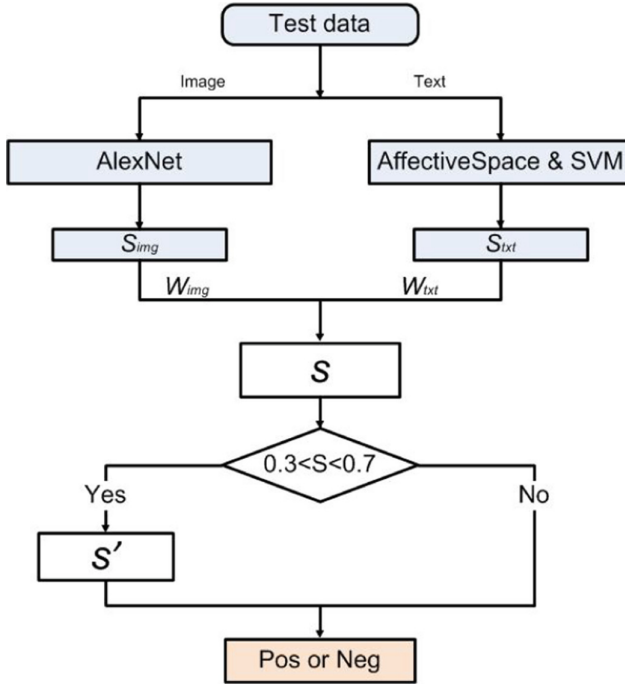


Fig. 2. Framework for TISC

4.3 TISC Model

Sentiment polarity prediction is based on features extracted from images and texts respectively. In order to balance visual features and textual features, our Text-Image Sentiment Classification (TISC) model employs the following computational approach to obtain sentiment scores of test data. Figure 2 is the flowchart of computing sentiment scores. First of all, we define the preliminary weight of our image features extraction architecture as:

$$w_{img} = \frac{Acc_{img}}{Acc_{img} + Acc_{txt}} \quad (4)$$

where Acc_{img} and Acc_{txt} is the accuracy for validation of image and text data respectively. Similarly, the textual weight is $w_{txt} = 1 - w_{img}$.

The preliminary sentiment score s is calculated by the following equation.

$$s = s_{img}w_{img} + s_{txt}w_{txt} \quad (5)$$

where s_{img} is the sentiment score for test image predicted by CNN, while s_{txt} is the score for text data given from AffectiveSpace 2.

For test data with $s \in [0.3, 0.7]$, we assume that such data do not have strong sentiment polarities or there is conflict between textual and visual features.

Hence, using the weighted scores to classify the sentiment polarities of these data is not appropriate. We import a new measure s' with variables $\alpha, \beta \in [0, 1]$ giving weights to visual and textual system respectively:

$$s' = 1 - (\alpha s_{img} + \beta s_{txt}) \quad (6)$$

With the sentiment scores measured as above, we define the sentiment polarity with 1 for *Positive* and 0 for *Negative* calculated by the following equation:

$$Polarity = \begin{cases} 1 & \text{with } s \geq 0.7 \text{ or } s' \geq 0.5 \\ 0 & \text{with } s \leq 0.3 \text{ or } s' < 0.5 \end{cases} \quad (7)$$

5 Experiments and Evaluation

In this section we first introduce the feature extraction from images for sentiment classification. Next, we compare the accuracy of TISC model with baselines for image sentiment analysis.

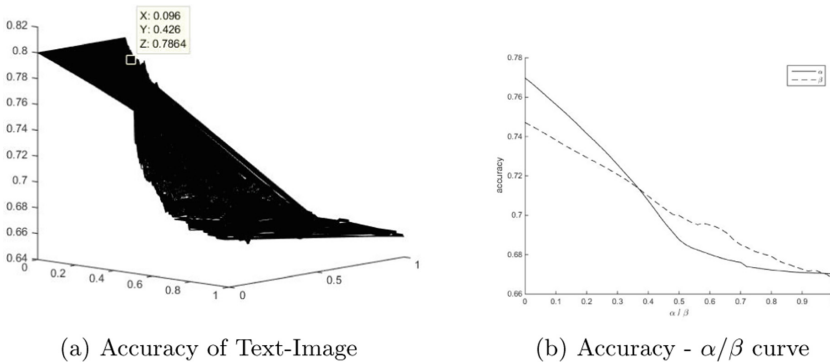


Fig. 3. Accuracy trend with α and β

5.1 Datasets

To conduct experiments on textual and visual features, we fine-tune AlexNet with 1269 labeled Twitter images¹. These images are annotated by 5 Amazon Mechanical Turk workers and we choose the images with the same sentiment label given by all the 5 workers (581 positive and 301 negative images). Since the data are unbalanced, in order to improve the fine-tuning on AlexNet, we double the negative images and increase the size of dataset to 18928 by adding rotations and reversals of each image. To judge our model, we test it on 596 images (463 positive and 133 negative images) with captions from Twitter [5].

¹ <http://www.cs.rochester.edu/u/qyou/DeepSent/deepsentiment.html>.

5.2 Experiment Results

In the first step of our approach, we fine tune the AlexNet with Flickr dataset to obtain the visual kernel features. Next, textual features are extracted by 5-fold validation using SVM classifier. To find the optimal combination of textual features and visual features, we use trial and error method, we progressively change all the parameters $\alpha, \beta \in [0, 1]$ in step of 0.002 (251,001 pairs of α and β). The highest accuracy in Fig. 3(a) reached 0.8051 and the accuracy is stable in the left part. Figure 3(b) shows how the value of α or β effects the accuracy and for each curve, the accuracy of α is the average accuracy for β on each value of α , which is similar to β . Table 1 shows the results of sentiment prediction using different methods and TISC using diverse parameters.

Table 1. Results of different methods

Method	α	β	Pred_neg	Pred_pos	Rec_neg	Rec_pos	Accuracy
Visual feature	–	–	0.3878	0.8352	0.4385	0.8043	0.7169
Textual feature	–	–	0.4122	0.8439	0.4692	0.8109	0.7356
TISC(1)	0.054	0.448	0.6596	0.8177	0.2385	0.9652	0.8051
TISC(2)	0.284	0.244	0.5574	0.8185	0.2615	0.9413	0.7915
TISC(3)	0.030	0.578	0.4787	0.8286	0.3462	0.8935	0.7729
TISC(4)	0.300	0.340	0.4058	0.8371	0.4308	0.8217	0.7356
TISC(5)	0.792	0.798	0.3610	0.8768	0.6692	0.6652	0.6661

Table 2. Comparison of other methods evaluated by AUC

Method	AUC
Low-level Features [5]	0.528
SentiBank [4]	0.514
TISC	0.586

5.3 Evaluation

Figure 3 shows the accuracy of TISC corresponding to different α and β . For example, the accuracy achieves 0.7864 at point (0.096, 0.426). From Fig. 3(a), it is shown that for α, β , if $\alpha + \beta \in (0, 0.5)$, then the results of system are stable with accuracy of 0.8051. However, from Eq. (6) reveals the truth that a high accuracy is with a low recall for Negative data since when $\alpha + \beta < 0.5$, for all data there is $s' > 0.5$ and then be classified as positive data. Figure 3(b) is the trend curves for accuracy- α/β from where we can see that in $[0, 1]$, the higher the values of α/β is, the lower the accuracy is.

From Table 1, the last five rows are the results reflecting to different (α, β) pairs with $\alpha + \beta > 0.5$, we can find that the TISC has a significant increase in

sentiment prediction compared with using single measures to predict sentiment polarity of test data and precision and recall are negative correlated. Table 2 shows that our model outperforms baselines by almost 6% in AUC. We compare with two baselines: Low-level Features are a set of features that can be useful for characterizing sentiment clues such as scenes, textures, and faces as well as other abstract concepts [5]; SentiBank is a concept representation with detectors trained on Flickr images.

6 Conclusion

This paper considers the application of Twitter images with captions for the prediction of sentiments applying fine-tune techniques. The Twitter images have corresponding labels or tweets, hence the merging of features from images and text is proposed. In this way, we can predict image sentiment as positive or negative with better performance. We see that the accuracy after fusing text and image features is higher than using a single modality. To extract the image features we consider AlexNet, which is a previously trained deep convolutional architecture. For text features we extract the significant concepts and project them on the AffectiveSpace of emotions. Lastly, we propose a novel sentiment score to combine the prediction from image and text features.

References

1. Achlioptas, D.: Database-friendly random projections: Johnson-lindenstrauss with binary coins. *J. comput. Syst. Sci.* **66**(4), 671–687 (2003)
2. Balduzzi, D.: Randomized co-training: from cortical neurons to machine learning and back again. arXiv preprint [arXiv:1310.6536](https://arxiv.org/abs/1310.6536) (2013)
3. Bingham, E., Mannila, H.: Random projection in dimensionality reduction: applications to image and text data. In: *ACM SIGKDD ACM*, pp. 245–250 (2001)
4. Borth, D., Chen, T., Ji, R., Chang, S.F.: SentiBank: large-scale ontology and classifiers for detecting sentiment and emotions in visual content. In: *Proceedings of the 21st ACM International Conference on Multimedia*, pp. 459–460. ACM (2013)
5. Borth, D., Ji, R., Chen, T., Breuel, T., Chang, S.F.: Large-scale visual sentiment ontology and detectors using adjective noun pairs. In: *Proceedings of the 21st ACM International Conference on Multimedia*, pp. 223–232. ACM (2013)
6. Cambria, E., Fu, J., Bisio, F., Poria, S.: AffectiveSpace 2: enabling affective intuition for concept-level sentiment analysis. In: *AAA*, vol. I, pp. 508–514 (2015)
7. Cambria, E., Hussain, A.: Sentic album: Content-, concept-, and context-based online personal photo management system. *Cogn. Comput.* **4**(4), 477–496 (2012)
8. Cambria, E., Poria, S., Bisio, F., Bajpai, R., Chaturvedi, I.: The CLSA model: a novel framework for concept-level sentiment analysis. In: Gelbukh, A. (ed.) *CICLing 2015*. LNCS, vol. 9042, pp. 3–22. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-18117-2_1
9. Cambria, E., Song, Y., Wang, H., Howard, N.: Semantic multi-dimensional scaling for open-domain sentiment analysis. *IEEE Intell. Syst.* **29**(2), 44–51 (2014)
10. Cambria, E., Wang, H., White, B.: Guest editorial: big social data analysis. *Knowl.-Based Syst.* **69**, 1–2 (2014)

11. Campos, V., Salvador, A., Giro-i Nieto, X., Jou, B.: Diving deep into sentiment: understanding fine-tuned CNNs for visual sentiment prediction. In: Proceedings of the 1st International Workshop on Affect & Sentiment in Multimedia, pp. 57–62. ACM (2015)
12. Chaturvedi, I., Satapathy, R., Cavallari, S., Cambria, E.: Fuzzy commonsense reasoning for multimodal sentiment analysis. *Pattern Recognit. Lett.* **125**, 264–270 (2019)
13. Chen, T., Borth, D., Darrell, T., Chang, S.F.: DeepSentiBank: visual sentiment concept classification with deep convolutional neural networks. arXiv preprint [arXiv:1410.8586](https://arxiv.org/abs/1410.8586) (2014)
14. Decherchi, S., Gastaldo, P., Zunino, R., Cambria, E., Redi, J.: Circular-ELM for the reduced-reference assessment of perceived image quality. *Neurocomputing* **102**, 78–89 (2013)
15. Grassi, M., Cambria, E., Hussain, A., Piazza, F.: Sentic web: a new paradigm for managing social media affective information. *Cogn. Comput.* **3**(3), 480–489 (2011)
16. Jia, Y., et al.: Caffe: convolutional architecture for fast feature embedding. In: Proceedings of the 22nd ACM International Conference on Multimedia, pp. 675–678. ACM (2014)
17. Katsurai, M., Satoh, S.: Image sentiment analysis using latent correlations among visual, textual, and sentiment views. In: 2016 IEEE International Conference on IEEE Acoustics, Speech and Signal Processing (ICASSP), pp. 2837–2841 (2016)
18. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: *Advances in Neural Information Processing Systems*, vol. 1, pp. 1097–1105 (2012)
19. Liu, B., Dai, Y., Li, X., Lee, W.S., Yu, P.S.: Building text classifiers using positive and unlabeled examples. In: Third IEEE International Conference on IEEE Data Mining, 2003. ICDM 2003, pp. 179–186 (2003)
20. Liu, H., Singh, P.: ConceptNet—a practical commonsense reasoning tool-kit. *BT Technol. J.* **22**(4), 211–226 (2004)
21. Lu, Y., Dhillon, P., Foster, D.P., Ungar, L.: Faster ridge regression via the subsampled randomized hadamard transform. In: *Advances in Neural Information Processing Systems*, vol. 26, pp. 369–377 (2013)
22. Narihira, T., Borth, D., Yu, S.X., Ni, K., Darrell, T.: Mapping images to sentiment adjective noun pairs with factorized neural nets. arXiv preprint [arXiv:1511.06838](https://arxiv.org/abs/1511.06838) (2015)
23. Plutchik, R.: Plutchik’s wheel of emotions (1980)
24. Qian, C., Chaturvedi, I., Poria, S., Cambria, E., Malandri, L.: Learning visual concepts in images using temporal convolutional networks. In: 2018 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 1280–1284 (2018)
25. Sarlos, T.: Improved approximation algorithms for large matrices via random projections. In: 47th Annual IEEE Symposium on IEEE Foundations of Computer Science, 2006. FOCS 2006, pp. 143–152 (2006)
26. Siersdorfer, S., Minack, E., Deng, F., Hare, J.: Analyzing and predicting sentiment of images on the social web. In: Proceedings of the 18th ACM International Conference on Multimedia, pp. 715–718. ACM (2010)
27. Strapparava, C., Valitutti, A.: Wordnet affect: an affective extension of wordnet. *LREC.* **4**, 1083–1086 (2004)
28. Susanto, Y., Livingstone, A., Ng, B.C., Cambria, E.: The hourglass model revisited. *IEEE Intell. Syst.* **35**(5), 96–102 (2020)
29. Tropp, J.A.: Improved analysis of the subsampled randomized hadamard transform. *Adv. Adapt. Data Anal.* (01n02) **3**, 115–126 (2011)

30. You, Q., Luo, J., Jin, H., Yang, J.: Joint visual-textual sentiment analysis with deep neural networks. In: Proceedings of the 23rd ACM International Conference on Multimedia, pp. 1071–1074. ACM (2015)
31. You, Q., Luo, J., Jin, H., Yang, J.: Robust image sentiment analysis using progressively trained and domain transferred deep networks. In: AAA, vol. I, pp. 381–388 (2015)
32. Zhang, Y., Shang, L., Jia, X.: Sentiment analysis on microblogging by integrating text and image features. In: Cao, T., Lim, E.-P., Zhou, Z.-H., Ho, T.-B., Cheung, D., Motoda, H. (eds.) PAKDD 2015. LNCS (LNAI), vol. 9078, pp. 52–63. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-18032-8_5



A Study on Far-Field Emotion Recognition Based on Deep Convolutional Neural Networks

Panikos Heracleous¹, Yasser Mohammad^{2,4}(✉), Koichi Takai¹, Keiji Yasuda¹, Akio Yoneyama¹, and Fumiaki Sugaya^{1,3}

¹ KDDI Research, Inc., 2-1-15 Ohara, Fujimino-shi, Saitama 356-8502, Japan
{pa-heracleous, ko-takai, ke-yasuda, yoneyama}@kddi-research.jp

² National Institute for Advanced Industrial Science and Technology, Tokyo, Japan
y.mohammad@aist.go.jp

³ MINDWORD Inc., 7-19-11 Nishishinjuku, Shinjuku-ku, Tokyo 160-0023, Japan
fsugaya@mindword.jp

⁴ Assiut University, Asyut, Egypt
yasserm@aun.edu.eg

Abstract. Automatic recognition of human emotions is a relatively new field, and is attracting significant attention in research and development areas because of the major contribution it could make to real applications. The current study focuses on far-field speech emotion recognition using the state-of-the-art spontaneous IEMOCAP emotional data. For classification, a method based on deep convolutional neural networks (DCNN) and extremely randomized trees is proposed. The method is also compared to support vector machines (SVM) and probabilistic linear discriminant analysis (PLDA) classifiers in the i-vector paradigm. When reverberant speech was classified using the proposed method, the classification rates were comparable to those obtained when using clean data. In the case of PLDA and SVM classifiers, the classification rates were significantly decreased. To further improve the performance of far-field speech emotion recognition, a method based on multi-style training is proposed, which results in significant improvements in the classification rates.

Keywords: Speech emotion recognition · Far-field · Deep convolutional neural networks · i-vectors · Multi-style training approach

1 Introduction

Emotion recognition plays an important role in human-machine communication [4]. Emotion recognition can be used in human-robot communication, when robots communicate with humans in accord with the detected human emotions, and also has an important role in call centers to detect the caller's emotional state in cases of emergency (e.g., hospitals, police stations), or to identify the level of

the customer's satisfaction (i.e., providing feedback). In the current study, emotion recognition based on speech in clean, noisy, and reverberant environments is experimentally investigated.

Previous studies reported automatic speech emotion recognition using Gaussian mixture models (GMMs) [34], hidden Markov models (HMM) [30], support vector machines (SVM) [22], neural networks (NN) [21], and deep neural networks (DNN) [31]. In [17, 38], speech emotion recognition using i-vector features and SVM is described. In [37], a study based on concatenated i-vectors is reported. Audiovisual emotion recognition is presented in [18]. The majority of studies dealing with speech emotion recognition address the problem using acted speech recorded with a close-talking microphone in a clean environment. In fact, only a few studies have used spontaneous or elicited emotion data. For real-world application, however, noise and reverberation in speech emotion recognition must also be addressed and analyzed. Previous studies [10, 29] have investigated the noise issue in speech emotion recognition by using data with superimposed white noise. In [35], several kinds of noise were recorded and superimposed onto clean data to simulate noisy emotional speech data, and adaptive noise cancellation was used as front-end to speech emotion recognizer. In [9], robust speech emotion recognition using a denoising autoencoder was used. In [23], spectral and cepstral audio denoising techniques were applied in speech emotion recognition.

The current study focuses on far-field speech emotion recognition, when the speaker is assumed to be located far way from the microphone and the speech emotion recognition system is being used in a hands-free mode. In such cases, the speech signal is also contaminated with environmental noise and reverberations. In the current study, the reverberant environments were simulated using real impulse responses recorded in rooms of different reverberation times, and convoluted with clean speech. Furthermore, real room noise was superimposed onto the reverberant data to simulate a more realistic situation. Instead of acted emotional speech, the state-of-the-art spontaneous emotional speech database IEMOCAP [3] was used. For classification, deep convolutional neural networks (DCNN) [1, 15] along with extremely randomized trees [8], multi-class SVM [5], and probabilistic linear discriminant analysis (PLDA) [13] classifiers were used. In the case of SVM and PLDA, i-vectors extracted from spectral features were used. Due to the limited amount of training data, i-vectors were not applied in the case of using DCNN. Instead, mel-frequency cepstral coefficients (MFCCs) [28] along with shifted delta cepstra (SDC) coefficients [2, 33] were used to extract informative features. The extracted features were then used by extremely randomized trees for emotion classification.

To improve robustness against noise and reverberation, a method based on multi-style training [24] is proposed. The authors were interested in whether i-vectors can capture the multiple noise and reverberation variability in the case of multi-style training, and how extremely randomized trees, SVM, PLDA with multi-style training perform in the case of far-field speech emotion recognition. Multi-style training is widely used in automatic speech recognition, and in the current study is adapted to speech emotion recognition. Specifically, the training

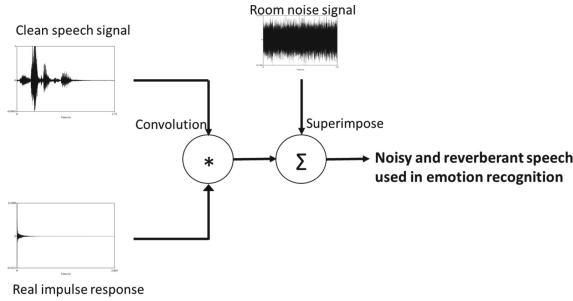


Fig. 1. Producing noisy and reverberant data using real impulse responses and real room noise.

data consist of data for different reverberation times, and do not include the same reverberation as the test data (i.e. reverberation-independent). In the current study, the proposed multi-style training is applied in all stages of the i-vector extraction, and not only for training the universal background model (UBM) and to compute the T total variability matrix.

2 Methods

2.1 Data

In the current study, the Interactive Emotional Dyadic Motion Capture (IEMOCAP) spontaneous emotional databases is used. The IEMOCAP database is an acted, multimodal and multispeaker database, collected at the SAIL lab of the University of Southern California, and it contains 12 h of audiovisual data produced by ten actors. Specifically, the IEMOCAP database includes video, speech, motion capture of face, and text transcriptions. It consists of dyadic sessions where actors perform improvisations or scripted scenarios, specifically selected to elicit emotional expressions. The IEMOCAP database is annotated by multiple annotators into several categorical labels, such as anger, happiness, sadness, neutrality, as well as dimensional labels such as valence, activation and dominance. In the current study, categorical labels are being used to classify the emotional states of neutral, happiness, anger, and sadness. To avoid unbalanced data, for training, 250 utterances and for testing 70 utterances randomly selected from each emotion were used.

2.2 Reverberant and Noisy Data

The reverberant data are produced using impulse responses convoluted with the clean data. Instead of simulated impulse responses (e.g., using the image method), in the current study real impulse responses recorded in five rooms with different $T_{[60]}$ reverberation times are being used. For recording, a linear microphone array with 14 transducers located at 2.83 cm intervals is used [20].

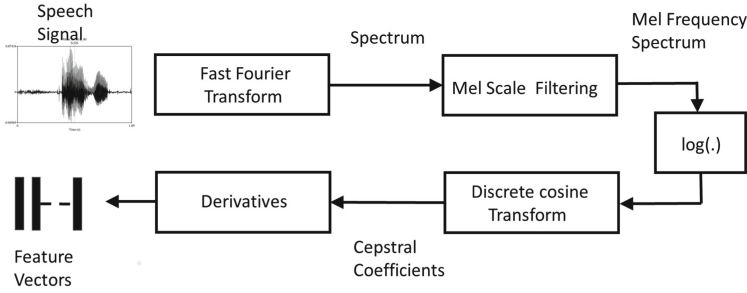


Fig. 2. Extraction of Mel-frequency Cepstral Coefficients (MFCC).

The impulse response is measured using the TSP method [32]. TSP length is 65536 points. The number of synchronous additions is 16. Impulse responses in five different rooms are being recorded. The $T_{[60]}$ reverberation times are 0.30, 0.47, 0.60, 0.78, and 1.3 s, respectively. The reverberant data are obtained using a convolution method to convolute the clean data with the impulse responses. Furthermore, real room noise (i.e., kitchen fan) at 20 dB signal-to-noise ratio (SNR) level is superimposed on the reverberant data to simulate a realistic noisy and reverberant speech emotion recognition environment. The reverberant data are produced using the following formula:

$$y(t) = x(t) * h(t) + n(t) \quad (1)$$

where $y(t)$ is the noisy and reverberant data, $x(t)$ is the close-talking speech data, $h(t)$ is the impulse response, and $n(t)$ is the additive noise. Figure 1 shows the method used to produce noisy and reverberant data.

2.3 Feature Selection

Mel-frequency cepstral coefficients (MFCCs) [28] are used in the experiments. MFCCs are very popular features in speech recognition, speaker recognition, emotion recognition, and language identification. Specifically, in the current study, 12 MFCCs plus energy are extracted each 10 ms using a window-length of 20 ms. Figure 2 shows the block diagram of MFCC extraction.

Shifted delta cepstral (SDC) coefficients have been successfully used in language recognition. In the current study, the use of SDC features in speech emotion recognition is also experimentally investigated in order to increase the temporal information in the feature vectors. The SDC features are obtained by concatenating delta cepstral across multiple frames. In this study, the SDC features are also used for speech emotion recognition, along with the MFCC features. The SDC features are described by four parameters, N , d , P and k , where N is the number of cepstral coefficients computed at each frame, d represents the time advance and delay for the delta computation, k is the number of blocks whose delta coefficients are concatenated to form the final feature vector, and

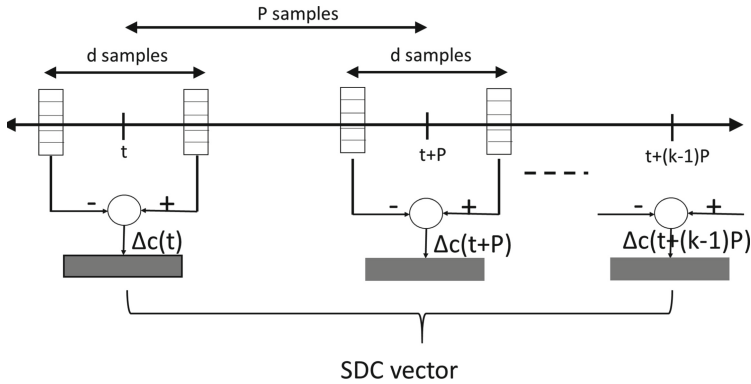


Fig. 3. Computation of SDC coefficients using MFCC and delta MFCC features.

P is the time shift between consecutive blocks. Accordingly, kN parameters are used for each SDC feature vector, as compared with $2N$ for conventional cepstral and delta-cepstral feature vectors. The SDC is calculated as follows:

$$\Delta c(t + iP) = c(t + iP + d) - c(t + iP - d) \tag{2}$$

The final vector at time t is given by the concatenation of all $\Delta c(t + iP)$ for all $0 \leq i < k - 1$, where $c(t)$ is the original feature value at time t . In the current study, the feature vectors with static MFCC features and SDC coefficients are of length 112. The concatenated MFCC and SDC features are used to extract the i-vectors that are used in classification when applying SVM and PLDA classifiers. In the case of using CNN, the MFCC and SDC features are used as input. Figure 3 illustrates the extraction of SDC features.

2.4 Evaluation Measures

In the current study, the classification rates are used as evaluation measures. The classification rate is defined as:

$$acc = \frac{1}{n} \sum_{k=1}^n \frac{No. \ of \ corrects \ for \ class \ k}{No. \ of \ trials \ for \ class \ k} \cdot 100 \tag{3}$$

where n is the number of the emotions.

2.5 The i-Vector Paradigm

A widely used classification approach in speaker recognition is based on GMMs with universal background models (UBM). In this approach, each speaker model is created by adapting the UBM using maximum a posteriori (MAP) adaptation. A GMM supervector is constructed by concatenating the means of the adapted

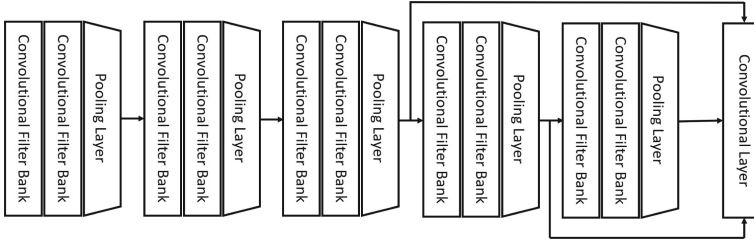


Fig. 4. The architecture of the deep feature extractor along with the classifier used during feature learning.

models. As in speaker recognition, GMM supervectors can also be used for emotion classification. The main disadvantage of GMM supervectors, however, is their high dimensionality, which imposes high computation and memory costs. In the i-vector paradigm, the limitations of high dimensional supervectors (i.e., concatenation of the means of GMMs) are overcome by modeling the variability contained in the supervectors with a small set of factors. Considering speech emotion classification, an input utterance can be modeled as:

$$\mathbf{M} = \mathbf{m} + \mathbf{T}\mathbf{w} \quad (4)$$

where \mathbf{M} is the emotion-dependent supervector, \mathbf{m} is the emotion-independent supervector, \mathbf{T} is the total variability matrix, and \mathbf{w} is the i-vector. Both the total variability matrix and emotion-independent supervector are estimated from the complete set of training data.

2.6 Classification Approaches

Convolutional Neural Networks (CNN). A deep neural network is a feed-forward neural network with more than one hidden layer. The units (i.e., neurons) of each hidden layer take all outputs of the lower layer and pass them through an activation function. A convolutional neural network is a special variant of the conventional network, which introduces a special network structure. This network structure consists of alternating convolution and pooling layers.

Convolutional neural networks have been successfully applied to sentence classification [14], image classification [26], facial expression recognition [12], and in speech emotion recognition [16]. Furthermore, in [7] bottleneck features extracted from CNN are used for robust language identification.

Deep learning (DL) is behind several of the most recent breakthroughs in computer vision, speech recognition, and agents that achieved human-level performance in several games like go, poker etc. In this paper, DL for learning informative features from the signal that is then used for emotion classification is investigated. The MFCC and SDC features are calculated using overlapping windows of length 20ms. This generates multidimensional time-series that represent the data for each session. The same train/test split is used in the following

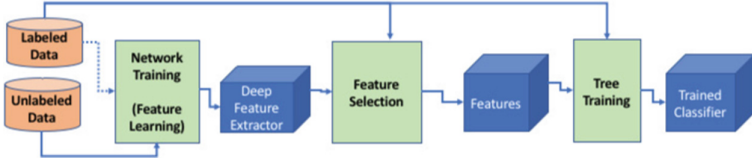


Fig. 5. The proposed training process showing the three stages of training and the output of each stage.

experiments as with the SVM and PLDA classifiers. The proposed method is a simplified version of the method recently proposed in [19] for activity recognition using mobile sensors.

The proposed classifier consists of a deep convolutional neural network (DCNN) followed by extremely randomized trees instead of the standard fully connected classifier. The motivation of using extremely randomized trees lies on previous observations showing the effectiveness in the case of small number of features. The network architecture is shown in the Fig. 4, and consists of a series of five blocks, each consisting of two convolutional layers (5×64) followed by a max-pooling layer ($width = 2$). Outputs from the last three blocks are then combined and flattened to represent the learned features.

Training of the classifier proceeds in three stages as shown in the Fig. 5: Network training, feature selection, and tree training. During network training, the deep convolutional neural network is trained with predefined windows of 21 feature MFCC/SDC blocks. Network training consists of two sub-stages: Firstly, the network is concatenated with its inverse to form an auto-encoder that is trained in unsupervised mode using all data in the training set and without the labels. Secondly, three fully connected layers are attached to the output of the network, and the whole combined architecture is trained as a classifier using the labeled training set. These fully connected layers are then removed, and the output of the neural network (i.e., deep feature extractor) represents the learned features.

The second training stage (i.e., feature selection) involves selecting few of the outputs from the deep feature extractor to be used in the final classification. Each feature (i.e., neuronal output i) is assigned a total *quality* ($Q(i)$) according to Eq. 5, where $\bar{I}_j(i)$ is z-score normalized feature *importance* ($I_j(i)$) according to a base feature selection method.

$$Q(i) = \sum_{j=0}^{n_f} w_j \bar{I}_j(i), \quad (5)$$

In the current study, three base selectors are utilized: randomized logistic regression [6], linear SVMs with L_1 penalty, and extremely randomized trees. Random linear regression (RLR) estimates feature importance by randomly selecting subsets of training samples and fitting them using a L_1 sparsity inducing penalty that is scaled for a random set of coefficients. The features that

appear repeatedly in such selections (i.e., with high coefficients) are assumed to be more *important*, and are given higher scores. The second base extractor uses a linear SVM with an L_1 penalty to fit the data and then select the features that have nonzero coefficients, or coefficients under a given threshold, from the fitted model. The third feature selector employs extremely randomized trees. During fitting decision trees, features that appear at lower depths are generally more important. By fitting several such trees, feature importance can be estimated as the average depth of each feature in the trees. Feature selection uses n -fold cross validation to select an appropriate number of neurons to keep in the final (fast) feature extractor (Fig. 5). For the sake of this work, the features (outputs) that have quality (Q_i) above the median value of qualities are kept.

Given the selected features from the previous step, an extremely randomized tree classifier is then trained using the labeled data set (i.e., tree training stage).

Note that the approach described above allows to generate a classification decision for each 21 MFCC/SDC blocks. To generate a single emotion prediction for each test sample, the outputs of the classifier need to be combined. One possibility is to use a recurrent neural network, an LSTM, or HMM to do this aggregation. Nevertheless, in this work the simplest voting aggregator, in which the label of the test file is the mode of the labels of all its data, is used.

Support Vector Machine (SVM). A support vector machine is a discriminative classifier, which is widely used in regression and classification. Given a set of labeled training samples, the algorithm finds the optimal hyperplane that categorizes new samples. SVM is among the most popular machine learning methods. The advantages of SVM include the support of high-dimensionality, memory efficiency, and versatility. However, when the number of features exceeds the number of samples, the SVM performs poorly. Another disadvantage is that SVM is not probabilistic because it works by categorizing objects based on the optimal hyperplane.

Probabilistic Linear Discriminant Analysis (PLDA). PLDA is a popular technique for dimension reduction using the Fisher criterion. Using PLDA, new axes are found, which maximize the discrimination between the different classes. PLDA was originally applied to face recognition [25], and is applied successfully to specify a generative model of the i-vector representation. PLDA was also used in speaker recognition. Adapting to emotion recognition, for the i -th emotion, the i -vector $\mathbf{w}_{i,j}$ representing the j -th recording can be formulated as:

$$\mathbf{w}_{i,j} = \beta + \mathbf{S}\mathbf{x}_i + \mathbf{e}_{i,j} \quad (6)$$

where β is a global offset (i.e., mean of training vectors), \mathbf{S} represents the between-emotion variability, and the latent variable \mathbf{x} is assumed to have a standard normal distribution, and to represent a particular emotion and channel. The residual term $\mathbf{e}_{i,j}$ represents the within-emotion variability, and it is assumed to have a normal distribution with zero mean and covariance Σ .

Table 1. Average classification rates using clean models and reverberant test data (IEMOCAP).

Reverberation time [sec]	Classification method		
	DCNN	SVM	PLDA
Clean	71.1	66.1	62.7
0.30	64.3	48.9	52.5
0.47	64.1	40.7	50.7
0.60	61.1	35.4	48.6
0.78	64.2	32.5	46.5
1.30	64.3	35.4	47.2

After the training and test i-vectors are computed, PLDA is used to decide whether two i-vectors belong to the same class. For this task, a test i-vector and an emotion i-vector are required. The emotion i-vectors are computed as the average of the training i-vectors, which belong to a specific emotion. A classification trial requires the emotion i-vectors, the test i-vector, and the PLDA model $\{\beta, \mathbf{S}, \Sigma\}$ parameters. A closed form for PLDA scoring is presented in [27].

3 Results

This section presents the results achieved for far-field speech emotion recognition using spontaneous emotional speech data. The results presented include classification rates when using clean and reverberant data, along with DCNN, SVM, and PLDA classifiers. Furthermore, the results when using the proposed multi-style training, which addresses far-field speech emotion recognition, are demonstrated. The i-vectors dimension was set to 100, and 128 Gaussian components were used in UBM training.

Using MFCC features along with SDC features and a DCNN classifier for the clean case of using IEMOCAP data, an average classification rate of 71.1% was obtained. This result is very promising and superior to the results obtained in similar studies [11,36]. The result also shows that SDC features can also be successfully used in speech emotion recognition. Based on this observation, in the following experiments, MFCC features concatenated with SDC coefficients are being used.

Table 1 shows the results when using clean training data and reverberant test data. As shown, in the case of SVM and PLDA classifiers and using clean models, the classification rates are decreased very significantly. On the other hand, when using DCNN, the effect of reverberation on classification rates is less significant. The results also show that PLDA classifiers outperforms SVM in the case of reverberant speech emotion recognition. On the other hand, using clean test data, SVM shows superior performance compared to PLDA. The highest rates, however, are being obtained when using DCNN.

Table 2. Average classification rates based on multi-style training and using reverberant test data (IEMOCAP).

Reverberation time [sec]	Classification method		
	DCNN	SVM	PLDA
0.30	68.3	66.4	60.6
0.47	68.4	66.4	61.3
0.60	70.4	66.1	61.3
0.78	71.0	66.1	63.0
1.30	71.1	65.4	60.6

Table 2 shows the classification rates when multi-style training along with reverberant test data were used in the case of IEMOCAP spontaneous emotional speech. As shown, multi-style training significantly improves the classification rates for all reverberation times. The achieved rates are almost the same as those obtained using clean data and in some cases higher rates are being obtained compared to the clean case (i.e., 0.30 s and 0.47 s reverberation times using SVM). A possible reason for this is the larger amount of training data used in creating the universal, reverberation-independent models. The results show the effectiveness of multi-style training in far-field speech emotion recognition, and demonstrate the ability of i-vectors to capture the variability of data with different reverberation times in a universal, reverberation-independent model set. As also shown in Table 2, the DCNN classifier has superior performance compared to SVM and PLDA in the case of speech emotion recognition based on a multi-style training approach. Specifically, the classification rates obtained are closely comparable to those obtained when using clean data.

These results support the statement, that multi-style training is an effective approach to deal with the decreased performance of a speech emotion recognition system operating in far-field mode. The results also justify the previous claims, that multi-style training can successfully be applied in the full extraction of i-vectors, and that i-vectors can capture the variability of data with a large number of reverberation times and additive noise.

4 Conclusion

The current study focused on far-field speech emotion recognition using the state-of-the-art IEMOCAP spontaneous emotional database based on DCNN. Using real impulse responses convoluted with clean data and real additive noise superimposed on the convoluted data, a realistic environment for far-field speech emotion recognition was produced. In the case of using clean IEMOCAP data

and DCNN, a 71.1% classification rate was obtained. This result is very promising and superior to other studies using the same databases. On the other hand, the rates were decreased in a reverberant environment when using SVM and PLDA classifiers in the i-vector paradigm. When using DCNN, the classification rates are comparable to those obtained when using clean data. To address the problem of reverberation and additive noise, a method based on multi-style training was proposed. This resulted in the classification rates being significantly improved. The results obtained are very promising and demonstrate the effectiveness of using DCNN in far-field speech emotion recognition. Furthermore, the results show that multi-style training can be successfully applied in far-field speech emotion recognition. The effectiveness of using SDC coefficients in speech emotion recognition was also demonstrated.

References

1. Abdel-Hamid, O., Mohamed, A.R., Jiang, H., Deng, L., Penn, G., Yu, D.: Convolutional neural networks for speech recognition. *IEEE/ACM Trans. Audio Speech Lang. Process.* **22**, 1533–1545 (2014)
2. Bielefeld, B.: Language identification using shifted delta cepstrum. In: Fourteenth Annual Speech Research Symposium (1994)
3. Busso, C., et al.: IEMOCAP: interactive emotional dyadic motion capture database. *J. Lang. Resour. Eval.* **42**, 335–359 (2008)
4. Busso, C., Bulut, M., Narayanan, S.: Toward effective automatic recognition systems of emotion in speech. In: Gratch, J., Marsella, S. (eds.) *Social Emotions in Nature and Artifact: Emotions in Human and Human-Computer Interaction*, pp. 110–127. Oxford University Press, New York (2013)
5. Cristianini, N., S-Taylor, J.: *Support Vector Machines*. Cambridge University Press, Cambridge (2000)
6. Friedman, J., Hastie, T., R.T.: Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *Ann. stat.* **28**(2), 337–407 (2000)
7. Ganapathy, S., Han, K., Thomas, S., Omar, M., Segbroeck, M.V., Narayanan, S.S.: Robust language identification using convolutional neural network features. In: *Proceedings of Interspeech* (2014)
8. Geurts, P., Ernst, D., Wehenkel, L.: Extremely randomized trees. *Mach. Learn.* **63**(1), 3–42 (2006)
9. Ha, H.K., Kim, N.K., Seong, W.K., Kim, H.K.: Noise-Robust Speech Emotion Recognition Using Denoising Autoencoder. *Audio Engineering Society Convention 140* (2016). <http://www.aes.org/e-lib/browse.cfm?elib=18164>
10. Huang, C., Chen, G., Yu, H., Bao, Y., Zhao, L.: Speech Emotion Recognition under White Noise. *Arch. Acoust.* **38**, 457–463 (2013)
11. Huang, C.W., Narayanan, S.: Attention assisted discover of sub-utterance in speech emotion recognition. In: *Proceedings of Interspeech*, pp. 1387–1391 (2016)
12. Huynh, X.-P., Tran, T.-D., Kim, Y.-G.: Convolutional neural network models for facial expression recognition using BU-3DFE database. In: *Information Science and Applications (ICISA) 2016*. LNEE, vol. 376, pp. 441–450. Springer, Singapore (2016). https://doi.org/10.1007/978-981-10-0557-2_44





13. Kanagasundaram, A., Dean, D., Sridharan, S.: Improving PLDA speaker verification with limited development data. In: Proceedings of 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1684–1688 (2014)
14. Kim, Y.: Convolutional neural networks for sentence classification. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1746–1751 (2014)
15. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Pereira, F., Burges, C.J.C., Bottou, L., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems 25, pp. 1097–1105. Curran Associates, Inc. (2012)
16. Lim, W., Jang, D., Lee, T.: Speech emotion recognition using convolutional and recurrent neural networks. In: Proceedings of Signal and Information Processing Association Annual Summit and Conference (APSIPA) (2016)
17. Liu, R.X.Y.: Using i-vector space model for emotion recognition. In: Proceedings of Interspeech, pp. 2227–2230 (2012)
18. Metallinou, A., Lee, S., Narayanan, S.: Decision level combination of multiple modalities for recognition and analysis of emotional expression. In: Proceedings of 2010 IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP, pp. 2462–2465 (2010)
19. Mohammad, Y., Matsumoto, K., Hoashi, K.: Deep feature learning and selection for activity recognition. In: Proceedings of the 33rd ACM/SIGAPP Symposium On Applied Computing, pp. 926–935. ACM SAC (2018)
20. Nakamura, S., Hiyane, K., Asano, F., Endo, T.: Sound Scene Data Collection in Real Acoustical Environments. *J. Acoust. Soc. Japan* **20**(3), 225–231 (1999)
21. Nicholson, J., Takahashi, K., Nakatsu, R.: Emotion recognition in speech using neural networks. *Neural Comput. Appl.* **9**(4), 290–296 (2000)
22. Pan, Y., Shen, P., Shen, L.: Speech emotion recognition using support vector machine. *Int. J. Smart Home* **6**(2), 101–108 (2012)
23. Pohjalainen, J., Ringeval, F., Zhang, Z., Schuller, B.: Spectral and cepstral audio noise reduction techniques in speech emotion recognition. In: Proceedings of ACM (2016)
24. Prabhavalkar, R., Alvarez, R., Parada, C., Nakkiran, P., Sainath, T.: Automatic gain control and multi-style training for robust small-footprint keyword spotting with deep neural networks. In: Proceedings of 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4704–4708 (2015)
25. Prince, S., Elder, J.: Probabilistic linear discriminant analysis for inferences about identity. In: Proceedings of International Conference on Computer Vision, pp. 1–8 (2007)
26. Rawat, W., Wang, Z.: Deep convolutional neural networks for image classification: a comprehensive review. *Neural Commun.* **29**, 2352–2449 (2017)
27. Romero, D.G., Wilson, C.E.: Analysis of i-vector length normalization in speaker recognition systems. In: Proceedings of INTERSPEECH, pp. 249–252 (2011)
28. Sahidullah, M., Saha, G.: Design, analysis and experimental evaluation of block based transformation in MFCC computation for speaker recognition. *Speech Commun.* **54**(4), 543–565 (2012)
29. Schuller, B., Arsic, D., Wallhoff, F., Rigoll, G.: Emotion recognition in the noise applying large acoustic feature sets. In: Proceedings of 3rd International Conference on Speech Prosody, pp. 276–279 (2006)

30. Schuller, B., Rigoll, G., Lang, M.: Hidden markov model-based speech emotion recognition. In: Proceedings of the 2003 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP). **I**, 401–404 (2003)
31. Stuhlsatz, A., Meyer, C., Eyben, F., Zielke, T., Meier, G., Schuller, B.: Deep neural networks for acoustic emotion recognition: raising the benchmarks. In: Proceedings of 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 5688–5691 (2011)
32. Suzuki, Y., Asano, F., Kim, H., Sone, T.: An optimum computer-generated pulse signal suitable for the measurement of very long impulse responses. *J. Acoust. Soc. Am.* **97**(2), 1119–1123 (1995)
33. T.-Carrasquillo, P., et al.: Approaches to language identification using gaussian mixture models and shifted delta cepstral features. In: Proceedings of ICSLP2002-INTERSPEECH2002, pp. 16–20 (2002)
34. Tang, H., Chu, S., Johnson, M.H.: Emotion recognition from speech via boosted gaussian mixture models. In: Proceedings of 2009 IEEE International Conference on Multimedia and Expo (ICME), pp. 294–297 (2009)
35. Tawari, A., Trivedi, M.: Speech emotion analysis in noisy real-world environment. In: Proceedings of International Conference on Pattern Recognition, pp. 4605–4608 (2010)
36. Tzinis, E., Potamianos, A.: Segment-based emotion recognition using recurrent neural networks. In: Proceedings of 2017 Seventh International Conference on Affective Computing and Intelligent Interaction (ACII), pp. 190–195 (2017)
37. Xia, R., Liu, Y.: Using i-vector space model for emotion recognition. In: Proceedings of INTERSPEECH, pp. 2227–2230 (2012)
38. Zhang, T., Wu, J.: Speech emotion recognition with i-vector feature and RNN model. In: 2015 IEEE China Summit and International Conference on Signal and Information Processing (ChinaSIP) (2015)

Syntax and Parsing



Cross-Framework Evaluation for Portuguese POS Taggers and Parsers

Sandra Collovini¹ , Henrique D. P. Santos¹ , Thiago Lima¹,
Evandro Fonseca¹, Bolivar Pereira¹, Marlo Souza² , Sílvia Moraes¹,
and Renata Vieira^{1,3} 

¹ Pontifícia Universidade Católica do Rio Grande do Sul, Porto Alegre, RS, Brazil
{sandra.abreu,henrique.santos.003,thiago.lima.001,evandro.fonseca,

bolivar.pereira}@acad.pucrs.br, silvia.moraes@pucrs.br

² Universidade Federal da Bahia, Salvador, BA, Brazil

msouza1@ufba.br

³ CIDEHUS, Universidade de Évora, Évora, Portugal

renatav@uevora.pt

Abstract. This work compares POS and parsing systems for the Portuguese language. We analyse available features, tagsets, and compare the results of POS tagging, and syntactic structure identification by means of both intrinsic and extrinsic evaluation methods. For such, we use in this work well-known metric for parser evaluation such as bracket cross, leaf ancestor for intrinsic evaluation, as well as the application of such parsers to the task of noun phrase identification, for extrinsic evaluation. The comparison proposed in this work takes into account the different linguistic theories and frameworks each parser subscribes to, but it is not dependent of any particular one.

Keywords: Parser evaluation · Portuguese parsers · Portuguese POS

1 Introduction

Parsing technology has increased greatly in the last decades, giving rise to a number of robust automatic parsers available in the field. Particularly, the rise of statistical parsers allied with machine learning methods for syntactic structure prediction, allowed the easy construction of automatic parsers based on annotated treebanks. With the proliferation of such tools, however, the problem of comparing their results have become apparent, a task commonly known as parser evaluation.

Such an evaluation is, in fact, not a simple task. The reason for this is that the parsers (and the treebanks used to construct them) are usually based on competing linguistic theories and frameworks, and the syntactic structure of a same sentence can diverge significantly according to each framework. As such, while one can easily apply well-established methodologies and metrics to compare parsers constructed over the same grammar/treebank, it is still an open problem how to compare two parser based on different linguistic theories or domains.

Recent research has been conducted on this problem, known as parser evaluation across linguistic domains and frameworks [4, 7, 19, 20, 27]. It is not clear yet in the literature, however, which methodology is more suitable for such a comparison.

Authors such as Carroll et al. [7], among others, have proposed that the use of representations such as Grammatical Relations (GR) are more adequate for parser evaluation than the use of tree structure. On the other hand, authors such as Mollá and Hutchinson [15] or Yuret *et al.* [27] propose that an extrinsic evaluation can provide a better way to evaluate parsing systems than intrinsic evaluations such as Grammatical Relations or syntactic tree comparison. This work acknowledges such challenges and brings some alternatives to compare parsing systems for the Portuguese language, independently of the linguistic framework to which each parser subscribes to.

This work is organized as follows: In Sect. 2, we present the linguistic Portuguese tools that will be compared. In Sect. 3, we describe the evaluation methods used, The Bosque Treebank, used as reference for the evaluation, and results are presented. Finally, Sect. 4 presents conclusions and future work.

2 Linguistic Annotation Tools

Linguistic annotation tools that provide information at the morpho-syntactic levels are extensively used as constituents of larger systems for many NLP tasks, and the quality of these annotations directly impacts the results of the bigger tasks. Currently, most syntactic parsers and POS taggers have been developed for the English language, with only a few available of them which can process texts in other languages, such as Portuguese. In the following, we present an overview of the morphosyntactic annotators available for the Portuguese language.

2.1 Part-Of-Speech Taggers

Part-Of-Speech (POS) tagging is an important preprocessing stage in NLP applications, and it is almost indispensable for any corpus research [10]. In order to analyze the sentence structure, for example, it is necessary to first recognize the grammatical categories of the words. Automatic POS tagger is a system responsible for identifying the grammatical category for each of the lexical items in a sentence.

In this section, we present the main, available, POS taggers for Portuguese. We analyze and compare their set of tags, since depending on the application for which the tagged text will be used, the number of tags can vary, and generally the quality of labeling directly impacts on applications performance. The POS tagger studied are presented below.

TreeTagger was developed by Helmut Schmid [22]. Besides Portuguese, it has been successfully used to tag many other languages. It can be used on any language if a lexicon and a manually tagged training corpus are available. It

is freely available for research, education and evaluation tasks. The Portuguese parameter file was provided by Pablo Gamallo.

NLTK the Natural Language Toolkit [3] is a suite of Python program modules, data sets and tutorials supporting research and teaching in computational linguistics and natural language processing. NLTK is written in Python and distributed under the GPL open source license. Over the past year the toolkit has been rewritten, simplifying many linguistic data structures and taking advantage of recent enhancements in the Python language.

NLPnet [9] is a tagger that was trained over a revision of Mac-Morpho [1], the biggest corpus of Portuguese text containing manually annotated POS tags. Many errors were corrected, yielding a much more reliable resource. We also trained a neural network based classifier for the POS tagging task, following an architecture that achieves state-of-the-art results in English.

UDPipe [26] performs tokenization, morphological analysis, part-of-speech tagging, lemmatization and dependency parsing for nearly all treebanks of Universal Dependencies, the latter is not available yet for the Portuguese language. In addition, the pipeline is easily trainable with training data in CoNLL-U format (and in some cases also with additional raw corpora) and requires minimal linguistic knowledge on the users' part. The training code is also released.

2.2 Syntactic Parsers

Syntactic parsers perform the structural analysis of phrases and their constituents. In this work we have analyzed the following parsers for the Portuguese language: Palavras, Freeling, CoGrOO, LX-Parser and MaltParser.

Palavras is an automatic tagger and parser for Portuguese that was developed by Eckhard Bick [2]. The formalism used follows the Constraint Grammar tradition (CG), introduced by Fred Karlsson [12]. PALAVRAS served as a model for the analysis of other languages in the VISL project (<http://visl.sdu.dk>), which is a core of tools and linguistic databases available for online use. Palavras provides the following information levels: morphological, POS, syntactic and dependency. This parser also has semantic prototype tags for nouns, proper nouns, verbs and some adjectives. An example output of Palavras is presented in Fig. 1, corresponding to sentence (1).

- (1) Fisher teria conhecido o ginecologista.
(Fisher would have known the gynecologist.)

Freeling is an open-source multilingual language processing library providing a wide range of analysis (morphological, named entity detection, PoS-tagging, parsing, Word Sense Disambiguation, Semantic Role Labelling, etc.) [18]. For Portuguese language, Freeling PT has the following functionalities available: morphological analysis, POS tagging, shallow parsing, named entity detection and classification. Figure 2 illustrates sentence (1) output provided by the Freeling PT. Freeling project was undertaken at the TALk research center to provide advances towards general availability of basic NLP tool and resources (<http://nlp.lsi.upc.edu/freeling/demo/demo.php>).

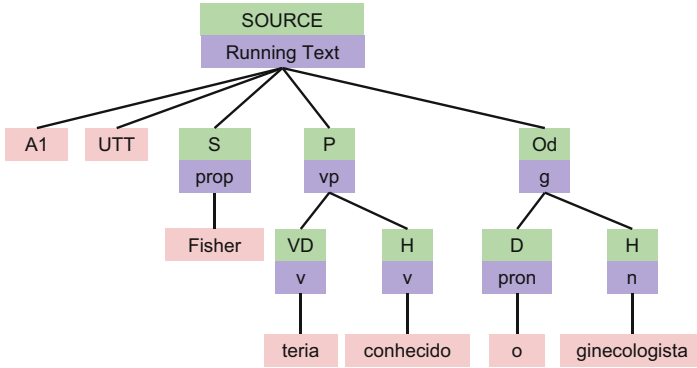


Fig. 1. Palavras syntactic tree (adapted)

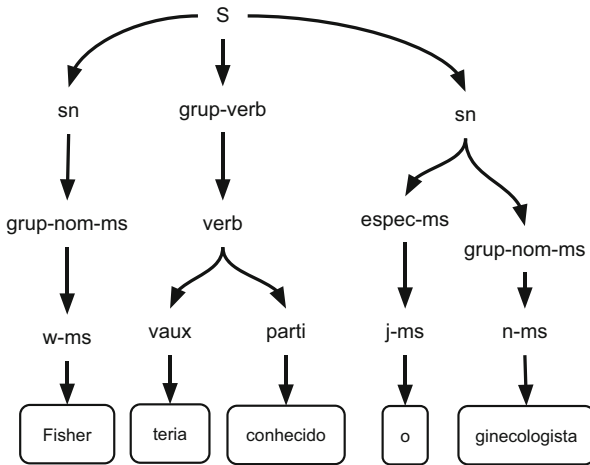


Fig. 2. Freeling syntactic tree (adapted)

CoGrOO [25] is an open-source grammar checker widely used for Portuguese. It is capable of identifying Portuguese grammatical errors such as pronoun placement, noun agreement, subject-verb agreement, usage of the accent stress marker, subject-verb agreement, and other common errors of Portuguese writing. Besides its use as a grammar checker, CoGrOO provides a set of linguistic annotation tools which can be used to process texts in the Portuguese language, such as a POS-taggers, chunkers and morphosyntactic annotators. In Fig. 3, we can see Cogroo’s output.

Sentence: Fisher teria conhecido o ginecologista
 Tokens:

Fisher	[]	prop	M=S
Teria	[ter]	v-fin	COND=3S
Conhecido	[conhecer]	v-ppc	M=S
O	[o]	art	M=S
ginecologista	[ginecologista]	n	F=S

 Chunks: [NP: Fisher]
 [VP: teria conhecido]
 [NP: o ginecologista]
 Shallow Structure: [SUBJ: Fisher]
 [P: teria conhecido]
 [ACC: o ginecologista]

Fig. 3. CoGrOO– Syntactic structures (adapted)

LX-Parser [24] is a robust parser for the Portuguese language freely available both as a web service and for download at <http://lxparser.di.fc.ul.pt/>. The current version of the LX-parser was built training a model for the version 1.6.5 of Stanford parser [13] using the CINTIL treebank [5]. The LX-Parser syntactic tree for the running example is depicted in Fig. 4.

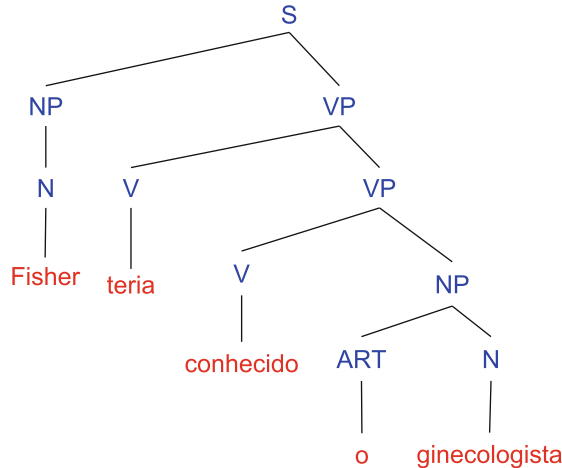


Fig. 4. LX-Parser syntactic tree (adapted)

MaltParser [17] is a language-independent system for dependency parser-generation which learns a deterministic dependency parser from a treebank. The parser-generator implements the arc-eager deterministic projective parser [16], which was trained in our work using a linear Support Vector Machine [11].

In this work, we trained the parser using the Universal Dependencies treebank for the Brazilian variant of the Portuguese language [14]. In order to use this

parser to annotate unseen text, we implemented a simple rule-based tokenizer for Portuguese and trained a POS tagger using the NLTK’s [3] implementation of the Brill Tagger [6].

3 Tools Evaluation

In this section, we present the Bosque Treebank, from which we extracted sentences used as reference for the evaluation. The evaluation methods used and the results achieved also are described.

3.1 Bosque Treebank

Bosque is a subset of treebank for Portuguese, “Floresta Sintá(c)tica”, composed of newspaper articles written in Brazilian and European Portuguese. Bosque has been automatically annotated by the Palavras parser and fully manually revised, with a current size of 9,368 sentences and 190,513 lexical units. In this work, we used version 7.4 (Brazilian Portuguese) in Constrain Grammar (CG) format [12] revised manually¹. Figures 5 and 6 present examples of sentence (1) in CG and PennTreebank format, respectively.

We extracted 10 reference sentences from Bosque for the evaluation of the taggers and parsers, and the selection criterion was sentence size equal to or greater than the average size of Bosque sentences.

Em	[em]	<sam->	PRP	@ADVL>	#1->7
essa	[esse]	<dem>	DET	F S @>N	#2->3
época	[época]		N	F S @P<	#3->1
,					#4->0
Fishel	[Fishel]		PROP	M S @SUBJ>	#5->6
teria	[ter]	<aux>	V COND	3S @STA	#6->0
conhecido	[conhecer]	<mv>	V PCP	@ICL-AUX<	#7->6
o	[o]	<artd>	ART	M S @>N	#8->9
ginecologista	[ginecologista]		N	F S @<ACC	#9->7
britânico	[britânico]		ADJ	M S @N<	#10->9
Robert=Winston	[Robert_Winston]		PROP	M S @N<	#11->9

Fig. 5. Sample in CG format

Since the Bosque treebank was constructed by manually correction of the output of the Palavras parser, we believe that we cannot evaluate the Palavras parser in a comparative manner while using this treebank. The reason for this is that, undeniably, the Bosque treebank reflects several theoretical and implementation decisions of the parser that would benefit Palavras in such an evaluation. The adopted evaluation methods are presented in the next Section.

¹ <http://www.linguateca.pt/floresta/corpus.html>.

```

(STA:fc1 (ADVL:pp (H:prp:em:: Em)
              (P<:np (>N:pron-det:esse:F_S::dem: essa)
                    (H:n:época:F_S::np-def: época)))
(,))
(SUBJ:np (H:prop:Fishel:M_S:: Fishel))
(P:vp (AUX:v-fin:ter:PR_3S_COND::: teria)
      (MV:v-pcp:conhecer::: conhecido))
(ACC:np (>N:art:o:M_S::artd: o)
        (H:n:ginecologista:F_S::np-def: ginecologista)
        (N<:adjp
              (H:adj:britânico:M_S::: britânico))
        (N<:np (H:prop:Robert_Winston:M_S::: Robert_Winston)
              ...

```

Fig. 6. Sample in PennTreebank format

3.2 Evaluation Methods

A variety of parser evaluation methods appear in the literature. A survey about the state-of-the-art in parser evaluation methodologies and metrics is presented in [7]. According to authors, the methods can be corpus-based or based on intrinsic properties of the parsers. The corpus-based methods are divided into those using annotated corpora and those using unannotated corpora. In this work, we apply some of these methods to evaluate the POS tagger and syntactic parser, which are described below.

General Comparison: We first present a general overview of the tools regarding linguistic information provided, in order of complexity, starting with the morphological and morphosyntactic levels (tokenization, lemmatization, POS tagging, morphology: gender, number, degree, person etc.); syntactic (shallow, full parsing, dependency); semantic and Named Entity (NE).

Part-of-Speech Assignment Accuracy: This measure computes the accuracy of POS tagger assignments of grammatical categories. Accuracy is used to evaluate POS tagger, due to the fact that there is a great deal of manually-corrected POS tagged data to use as test corpora. In literature, there are many ways to calculate the accuracy, here we computed, for each POS tagger, the number of correct words tagged, compared with the reference, divided by total number of words tagged. For this, the first step was to perform a mapping between the different POS tags of each tool in comparison to the reference. The reference sentences (from the Bosque corpus) follow the tagset of the Palavras parser.

Structural Consistency: To evaluate the result of the parsers, we analyzed the parsing results using well-known metrics for constituency parsing, such as bracket crossing [23] and leaf ancestor [21] metrics. It is of importance to notice two things in our evaluation. First, the results of all parsers were converted to the Penn Treebank format to perform such evaluation. The second is that we decided to evaluate only the structure of the trees, not whether the tags were

correctly labelled. Since these metrics are sensible to differences in tokenization, a step of manual tokenization correction was necessary to guarantee that the trees resulted by the parsers could be analyzed.

Four parsers were evaluated according to these metrics: CoGrOO, Freeling, LX-Parser and Malt Parser. From them, only the MaltParser - a dependency parser - returns an output not consistent with the Penn Treebank format which required transformation of the output. To perform this transformation, the following rules were applied: (i) the entire sentence is inside a constituent with the head as the root element of the syntax tree; (ii) for each token T , if there are tokens which depend on it in the syntax tree, then there is a constituent containing all the tokens which depend on T , for which the token T is the head.

Noun Phrases Assignment Accuracy: Current methods for evaluating the accuracy of syntactic parsers are based on measuring the degree to which parser output replicates the analyses assigned to sentences in a manually annotated corpus. In this work, we analyze the noun phrase (NP) contained in the 10 reference sentences compared the output parser: Cogroo, Freeling, LX-Parser, MaltParser. For this, we extract the noun phrases (the most external and also the internal ones) of both the reference and the outputs of each analyzed parser and we computed the accuracy. We consider the correct NP (when NP are equal at system output and at reference: accuracy = 1) and partially correct (when at least one NP token at system output (NPs) corresponds to a reference token (NPr): accuracy = $0.5 * (NPs / NPr)$ [8]).

3.3 Results

An overview of the features available for the Portuguese tools under analysis is illustrated in Table 1. We also present the language and terms/license that each tool was developed in. Most tools perform POS tagging; the annotation of lemmatization and morphological are performed by some of the tools: Palavras, Freeling, Cogroo, UDPipe and TreeTagger. There are two shallow parsers (Freeling and Cogroo), and Palavras is the only full parser. Palavras also provides semantic tags, dependency and Named Entity. It is noteworthy that the dependency information is provided also by MaltParser and Named Entities are annotated by Freeling.

Table 3 illustrates the mapping between the different POS taggers. We can see that Cogroo and NLTK use the same POS tags defined by the Palavras²; UDPipe and MaltParser use the Universal POS tags³ POS tags; Freeling and TreeTagger use the EAGLES⁴ POS tags; NLPnet use the tagset of Mac-Morpho; and LX-Parser use the tagset of CINTIL treebank [5]. One of the difficulties for the evaluation was the different attributes of each category considered by the POS taggers, for example, the category “verb” was considered by all the tools,

² <https://visl.sdu.dk/visl/pt/info/portsymbol.html>.

³ <http://universaldependencies.org/u/pos/>.

⁴ <http://www.ilc.cnr.it/EAGLES/browse.html>.

however their attributes differed among them, such as type (auxiliary verb, main verb) and genre (feminine, masculine, neuter), among others.

Table 1. Tools features comparison

	Palavras	Freeling	Cogroo	NLTK	NLPnet	UDPipe	MaltParser	LX-Parser	TreeTagger
Language	C, Perl	C++	Java	Python	Python	Python	Java	Java	C++
Terms/License	Proprietary	Opensource	Opensource	Opensource	Opensource	Opensource	Opensource	Opensource	Opensource
Tokenization	✓	✓	✓	✓	✓	✓		✓	✓
Lemmatization	✓	✓	✓		✓	✓			✓
PoS tagging	✓	✓	✓	✓	✓	✓		✓	✓
Morphological	✓	✓	✓			✓			✓
Shallow parsing		✓	✓						
Full parsing	✓							✓	
Dependency	✓						✓		
Semantic	✓								
Named Entity	✓	✓							

Table 2. Parser evaluation results

Tools	POS ↑	Bracket Crossing	Leaf Ancestor	NPs
Cogroo	98.81	50.68	88.15	68.08
Palavras	96.93	NE	NE	NE
TreeTagger	92.73	–	–	–
Freeling	91.39	31.78	88.10	53.75
UDPipe	91.24	–	–	–
LX-Parser	87.91	44.27	77.71	47.25
MaltParser	87.20	35.33	89.90	36.63
NLPnet	81.17	–	–	–
NLTK HMM	77.95	–	–	–
NLTK Def	72.35	–	–	–

After, we calculated the average accuracy of the sentences for each POS tagger and parser, and illustrated the results in Table 2. The best results were achieved by the Cogroo parser, with the best POS tagging performance, Crossing-bracket and noun phrase accuracy. As a open-source software Cogroo also has a large set of features, missing just Named Entity identification, that Freeling does. Palavras was not evaluated due to its influence in the reference generation (NE). MaltParser presented better performance for leaf ancestor metric. We believe the reason for the discrepancy between the results for MaltParser from one metric to the other is due to the fact that there is an impact on the transformation of the dependency structure to the phrasal structure, where the constituents are less structured than a constituency parser, however this metric is more flexible and the conversion did not impact the results.

Table 3. Portuguese tools and Bosque Treebank Tagset

Tagset	Bosque	Palavras	Freeling	Cogroo	NLTK	NLPnet	UDPipe	MaltParser	LX-Parser	TreeTagger
definite article	ART	<artd> DET	DA	art	art	ART	DET	DET	ART	DA
indefinite article	ART	<arti> DET	DI	art	art	ART	DET	DET	ART	DI
preposition	PRP	PRP	SP	prp	prp	PREP	ADP	ADP	P	S
noun	N	N	NC	n	n	N	NOUN	NOUN	N	N
proper noun	PROP	PROP	NP	prop	prop	NPROP	PROPN	PROPN	N	NP
personal pronoun	PERS	PERS	PP	pron-pers	pron-pers	PROPESS	PRON	PRON	PRS	PP
determiner pronoun	DET	DET	D*	pron-det	pron-det	PROADJ	DET	DET	DEM	D
independent pronoun	INDP	SPEC	PR	pron-indp	pron-indp	PRO-KS	PRON	PRON	CL	PI
adjective	ADJ	ADJ	A	adj	adj	ADJ	ADJ	ADJ	A	A
adverb	ADV	ADV	R	adv	adv	ADV	ADV	ADV	ADV	R
auxiliary verb	<aux> V	<aux> V	VA*	v-fin	v-fin	VAUX	AUX	AUX	V	VA
finite verb present	V PR	V PR	V*	v-fin	v-fin	V	VERB	VERB	V	VM
finite verb past perfect	V PS	V PS	V*	v-fin	v-fin	V	VERB	VERB	V	VM
finite verb past imperfect	V IMPF	V IMPF	V*	v-fin	v-fin	V	VERB	VERB	V	VM
infinitive verb	V INF	V INF	V*N	v-inf	v-inf	V	VERB	VERB	V	VMN
participle verb in compound tense	V PCP	V PCP	V*P	v-pcp	v-pcp	PCP	VERB	VERB	PPT	VM
participle verb not in compound tense	V PCP	V PCP	V*P	v-pcp	v-pcp	PCP	VERB	VERB	PPA	VM
gerund verb	V GER	V GER	V*G	v-ger	v-ger	V	VERB	VERB	GER	VMG
gerund as auxiliary verb									GERAUX	VAG
numeral	NUM	NUM	Z	num	num	NUM	NUM	NUM	CARD, ORD	Z
subordinating conjunction	KS	KS	CS	conj-s	conj-s	KS	SCONJ	SCONJ	C	CS
coordinating conjunction	KC	KC	CC	conj-c	conj-c	KC	CCONJ	CCONJ	CONJ	CC
interjection	IN	IN	I	intj	intj	IN	INTJ	INTJ		I

4 Conclusions and Future Work

In this work, we show the comparison of 9 annotation tools for (POS taggers and parsers) for the Portuguese language. A correlation Table presents the variety of tagsets used across the tools analysed in this work. To correlate different parsers is a difficult task, due to the variety of tags, different underlying assumptions and their structures. To overcome such difficulties, we employed several different evaluation methods and metrics in the literature and compared all the parser according to each of these metrics. As a result, considering the method adopted in this work, the Cogroo grammar-checker was the parser that presented a better overall performance.

As future work, we intend to perform the evaluation on the entirety of the Bosque treebank. This was not possible in this work since the comparison of the results required several steps of manual intervention. For example, in comparing the results of the POS Tagging task, it was required to map the possible equivalences of the tags in each tagsets and evaluate the resulting annotation of each sentence for all tools. Using the universal dependency tagset [14], however, for which such mappings have been provided, we believe we can automatize most of such labour-intensive analysis and provide a more comprehensive comparison of the tools.

We would also like to combine the current evaluation methodology with that of grammatical relations, by mapping each syntactic construction in the constituency parsers' tagsets to a comprehensive set of grammatical relations. We believe applying different evaluation strategies, we can realistically compare the results of syntactic parsers for the Portuguese language, despite their different linguistic foundations.

References

1. Aluísio, S., Pelizzoni, J., Marchi, A.R., de Oliveira, L., Manenti, R., Marquiafável, V.: An account of the challenge of tagging a reference corpus for brazilian portuguese. In: Mamede, N.J., Trancoso, I., Baptista, J., das Graças Volpe Nunes, M. (eds.) PROPOR 2003. LNCS (LNAI), vol. 2721, pp. 110–117. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-45011-4_17
2. Bick, E.: The Parsing System Palavras. Automatic Grammatical Analysis of Portuguese in a Constraint Grammar Framework. University of Aarhus, Aarhus (2000)
3. Bird, S.: NLTK: the natural language toolkit. In: Proceedings of the COLING/ACL on Interactive presentation sessions, pp. 69–72. Association for Computational Linguistics (2006)
4. Bos, J., et al. (eds.): Coling 2008: Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation. In: Association for Computational Linguistics (2008)
5. Branco, A., Silva, J., Costa, F., Castro, S.: Cintil treebank handbook: design options for the representation of syntactic constituency. Technical Report TR-2011-02, Universidade de Lisboa (2011)
6. Brill, E.: Transformation-based error-driven learning and natural language processing: a case study in part-of-speech tagging. *Comput. Linguist.* **21**(4), 543–565 (1995)
7. Carroll, J., Briscoe, T., Sanfilippo, A.: Parser evaluation: a survey and a new proposal. In: Proceedings of the 1st International Conference on Language Resources and Evaluation, pp. 447–454 (1998)
8. Diana, N.C., Santos, Seco, N.: Avaliação no harem: Métodos e medidas. In Diana Santos and Nuno Cardoso (eds.) Reconhecimento de entidades mencionadas em português: Documentação e actas do HAREM, a primeira avaliação conjunta na área. Linguateca. Departamento de Informática, Faculdade de Ciências da Universidade de Lisboa. DI-FCUL TR-06-17, Linguateca, november 2006 2007
9. Fonseca, E.R., Rosa, J.L.G.: Mac-morpho revisited: towards robust part-of-speech tagging. In: Proceedings of the 9th Brazilian Symposium in Information and Human Language Technology, pp. 98–107. sn (2013)
10. Giesbrecht, E., Evert, S.: Is part-of-speech tagging a solved task? an evaluation of POS taggers for the german web as corpus. In: Alegria, I., Leturia, I., Sharoff, S. (eds.) Proceedings of the 5th Web as Corpus Workshop (WAC5), San Sebastian, Spain (2009)
11. Hearst, M.A., Dumais, S.T., Osuna, E., Platt, J., Scholkopf, B.: Support vector machines. *IEEE Intell. Syst. Appl.* **13**(4), 18–28 (1998)
12. Karlsson, F., Voutilainen, A., Heikkilä, J., Anttila, A. (eds.): Constraint Grammar: A Language-Independent System for Parsing Unrestricted Text. Mouton de Gruyter, Berlin (1995)
13. Klein, D., Manning, C.D.: Fast exact inference with a factored model for natural language parsing. In: Advances in Neural Information Processing Systems, pp. 3–10 (2003)
14. McDonald, R.T., et al.: Universal dependency annotation for multilingual parsing. In: ACL, vol. 2, pp. 92–97 (2013)
15. Mollá, D., Hutchinson, B.: Intrinsic versus extrinsic evaluations of parsing systems. In: Proceedings of the EACL 2003 Workshop on Evaluation Initiatives in Natural Language Processing: Are Evaluation Methods, Metrics and Resources Reusable?, pp. 43–50. Association for Computational Linguistics (2003)

16. Nivre, J.: An efficient algorithm for projective dependency parsing. In: Proceedings of the 8th International Workshop on Parsing Technologies (IWPT). Citeseer (2003)
17. Nivre, J., et al.: MaltParser: a language-independent system for data-driven dependency parsing. *Nat. Lang. Eng.* **13**(2), 95–135 (2007)
18. Padró, L.; Stanilovsky, E.: FreeLing 3.0: towards wider multilinguality. In: Proceedings of the Language Resources and Evaluation Conference (LREC 2012), Istanbul, Turkey, ELRA (2012)
19. Preiss, J.: Using grammatical relations to compare parsers. In: Proceedings of the Tenth Conference on European Chapter of the Association for Computational Linguistics-Vol. 1, pp. 291–298. Association for Computational Linguistics (2003)
20. Sagae, K., Miyao, Y., Matsuzaki, T., Tsujii, J.: Challenges in mapping of syntactic representations for framework-independent parser evaluation. In: The Workshop on Automated Syntactic Annotations for Interoperable Language Resources (2008)
21. Sampson, G., Babarczy, A.: A test of the leaf-ancestor metric for parse accuracy. *Nat. Lang. Eng.* **9**(4), 365–380 (2003)
22. Schmid, H.: Probabilistic part-of-speech tagging using decision trees. In: Proceedings of International Conference on New Methods in Language Processing, Manchester, UK (1994)
23. Sekine, S., Collins, M.: Evalb bracket scoring program. <http://www.cs.nyu.edu/cs/projects/proteus/evalb> (1997)
24. Silva, J., Branco, A., Castro, S., Reis, R.: Out-of-the-box robust parsing of Portuguese. In: Pardo, T.A.S., Branco, A., Klautau, A., Vieira, R., de Lima, V.L.S. (eds.) PROPOR 2010. LNCS (LNAI), vol. 6001, pp. 75–85. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-12320-7_10
25. Silva, W.D.C.: Aprimorando o corretor gramatical cogroo. In Master’s Dissertation. Universidade de São Paulo (2013)
26. Straka, M., Hajic, J., Straková, J.: UDPipe: trainable pipeline for processing CoNLL-U files performing tokenization, morphological analysis, POS tagging and parsing. In: Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC) (2016)
27. Yuret, D., Han, A., Turgut, Z.: SemEval-2010 task 12: parser evaluation using textual entailments. In: Proceedings of the 5th International Workshop on Semantic Evaluation, pp. 51–56. Association for Computational Linguistics (2010)



Parsing Arabic with a Semi-automatically Generated TAG: Dealing with Linguistic Phenomena

Cherifa Ben Khelil¹(✉) , Chiraz Ben Othmane Zribi² , Denys Duchier³,
and Yannick Parmentier⁴

¹ LIFAT - IUT de Blois, Université de Tours, Blois, France
cherifa.bk@gmail.com

² RIADI - ENSI, Université La Manouba, La Manouba, Tunisia
chiraz.zribi@ensi-uma.tn

³ LIFO - Université d'Orléans, Orléans, France
denys.duchier@univ-orleans.fr

⁴ LORIA- Projet SYNALP - Université de Lorraine, Vandoeuvre-les-Nancy, France
yannick.parmentier@loria.fr

Abstract. Arabic is a challenging language when it comes to grammar production and parsing. It combines complex linguistic phenomena with a rich morphology that make its processing particularly ambiguous. This led us to choose the Tree-Adjoining Grammar (TAG) formalism. Indeed, TAG provides sufficient constraints for handling diverse linguistic phenomena and seems to be adequate to represent Arabic syntactic structures. In this paper, we present a semi-automatically generated TAG for modern standard Arabic using a compiler and a metagrammatical description language called XMG (eXtensible MetaGrammar). We describe the linguistic coverage of our grammar, and show how we used TAG and XMG's properties to define in an expressive and concise way different linguistic phenomena. To check the coverage of our grammar, we have set up a development environment including a parser and using a test corpus of linguistic phenomena gathering both grammatical and ungrammatical sentences.

Keywords: Tree adjoining grammar (TAG) · Metagrammar · Parsing · Corpus of linguistic phenomena · Arabic language

1 Introduction

Arabic is a challenging language when it comes to grammar production and parsing. It exhibits specific features such as a relatively free word order, combined with a rich morphology, and the omission of diacritics representing vowels in most of the written text. These linguistic phenomena affect the syntactic parsing process and makes it more difficult. Several methods adopted the rule-based approach for parsing modern standard Arabic (MSA). This approach uses

a well-defined formal grammar. Among these methods, some researchers used a Head-Driven Phrase Structure Grammar (HPSG) to represent Arabic syntax such as the parser MASPARE [1] and the platform PHARAS [2]. Others relied on the Lexical Functional Grammar (LFG) [3]. However, this approach has a lot of difficulties. In fact, grammars are usually encoded manually thus its construction process takes a lot of time [3]. In addition, it is difficult to have a grammar that covers all the syntactical structures of a language. In this context, many efforts have been put into semiautomatic grammar production, either by acquiring grammar rules from annotated corpora [4] or by using description languages to capture generalizations among these rules. The latter permits to formally specify the structures of a target grammar. This grammar specification (called metagrammar) can be compiled into an actual electronic grammar. In this work, we applied such grammar production techniques to the description of Arabic. We used the XMG description language [5] to semi-automatically generate a Tree-Adjoining Grammar (TAG) [6] for MSA called ArabTAG V2.0. Our choice was motivated by the power of representation of TAG (simple, complex, combinatorial, shared structures, etc.) and its ability to deal with certain phenomena that are recurrent in Arabic. In the following section, we present the grammatical resources used in the generation process of our grammar. Section 3 gives a concise description of the syntax of Arabic using XMG. In Sect. 4, we show how we used TAG and XMG's properties to deal with different linguistic phenomena of Arabic. In Sect. 5, we present a corpus of phenomena generated following the verification phase of our grammar. In Sect. 6, we comment on the ArabTAG grammar coverage. Finally in Sect. 7, we conclude and present some short-term perspectives of development of the grammar.

2 Semi-automatically Generating ArabTAG V2

Tree Adjoining Grammar (TAG) [7] is a syntactic formalism that takes into account the links between the constituents of the sentence to build grammatical representations. It consists of a set of elementary trees divided in :

- initial tree: it is a tree whose leaf nodes are either terminal symbols or non-terminal symbols. The non-terminal symbols are called substitution nodes and are marked with the symbol (\downarrow).
- auxiliary tree: it has also a leaf node labeled with a non-terminal symbol called “foot node” and is marked with the symbol (*). The foot node and the root of the auxiliary tree must be of the same category.

The two compositions operations authorized by TAG are substitution and adjunction. The resulting tree obtained by the end of these operations is called a derived tree. Substitution appends a frontier node with another tree whose top node has the same symbol. Adjunction is more powerful since it allows inserting an auxiliary tree into the center of another tree. We cannot, assert that this formalism is undoubtedly the best to represent Arabic. Nevertheless, its characteristics make it possible to represent syntactic structures and frequent

phenomena in Arabic. To our knowledge, there are very few TAG-based descriptions of Arabic. The first TAG by Habash and Rambow [4], was extracted from an Arabic Treebank (namely the Penn Arabic TreeBank - PATB). The corpus they used is the Part 1 v 2.0 of PATB [8,9]. The second is a handcrafted tree-adjoining grammar named ArabTAG (Arabic Tree Adjoining Grammar) [10]. Our work takes its origins from the latter. This grammar describes different syntactic components of different levels: sentences, phrases and words, as well as the various information related to them (morphological and syntactic information). ArabTAG has feature structure and is semi-lexicalized. It contains two sets of elementary trees: 35 lexicalized trees (reserved to prepositions, modifiers, conjunctions, demonstrative pronouns) and 215 patterns trees (represent verbs, nouns, adjectives or any kind of phrases). The construction of these structures was based on school grammar books and books of Arabic grammar [11]. The current version of this grammar has some limitations that can be summarized as follows:

- Minimal coverage of syntactic structures. Structures enriched with supplements (circumstantial complements of time, place, etc.) are not described.
- The representation of forms of agglutination is not well reflected.
- The grammar emphasizes syntactic relations without regard to semantic information.
- ArabTAG is not organized in a hierarchical way, which does not facilitate grammar extension and maintenance.

We have proposed a new version ArabTAG V2.0 [6] that takes into account the aspects mentioned above. We used XMG (eXtensible MetaGrammar) description language [5] to describe Arabic for it exhibits particularly pertinent features:

- it is highly expressive, since it defines highly factorized grammar descriptions.
- it is particularly adapted to the description of tree grammars and has been used to develop several electronic TAG grammars for e.g. French¹, English², German³.
- it is highly extensible and can be configured to describe various levels of language, such as semantic or morphology.

We have semi-automatically generated ArabTAG V 2.0 (see Fig. 4) from a reduced description of grammar rules. First, the metagrammatical language XMG is used to define TAG trees. It is described as (conjunctive and disjunctive) combinations of tree fragments. Such fragments are defined as formulas of a tree description logic based on dominance and precedence relations between node variables. Then, this compact description (namely meta-grammar) is automatically compiled into an actual electronic grammar ArabTAG V2.0 (coded in XML - Extensible Markup Language) by the XMG2⁴ compiler [12].

¹ <https://sourcup.renater.fr/xmg/frenchmetagrammar/index.html>.

² <http://homepages.inf.ed.ac.uk/s0896251/XMG-basedXTAG/titlepage.html>.

³ <http://www.sfs.uni-tuebingen.de/emmy/res.html>.

⁴ XMG2 extends XMG by including a meta metagrammar compiler.

3 Describing Basic Syntactic Structures of Arabic in ArabTAG V2

In this section, we aim to illustrate the new formulation of ArabTAG by focusing on the modelization of simple verb subcategorization (for verbal sentences) and nominal sentence. In order to make this presentation easier, we use a combination of logical and graphical notation rather than XMG's concrete syntax.

3.1 Verbal Sentence

We started by focusing on the modelization of simple verb subcategorization. We defined $\text{EpineVerbe}(C)$. It is an abstraction that contributes a fragment of tree description for the verbal spine of the MorphActive class. The purpose of adding AG and AD nodes will be explained later in this article. Nodes here are colored⁵ (represented by B, W and R for black, white and red respectively). EpineVerbe is parameterized by a color C as depicted below⁶:

$$\begin{array}{c} \text{SV}^C[\text{cat}=\text{sv}] \\ | \\ \text{AG}^C[\text{cat}=\text{adv}g] \\ | \\ \text{AD}^C[\text{cat}=\text{adv}d] \\ | \\ \text{V}_\diamond^C[\text{cat}=\text{v}] \end{array}$$

$\text{EpineVerbe}(C) \longrightarrow$

MorphActive contributes the actual verb spine (which can be seen as a resource) and therefore instantiates EpineVerbe with the color black:

$$\text{MorphActive} \longrightarrow \text{EpineVerbe}(B)$$

EpineArg is an abstraction used for attaching to the verbal spine a tree description for an argument.

$$\text{EpineArg} \longrightarrow [\text{AG}] \Leftarrow \text{EpineVerbe}(w) \wedge \text{AD}^R[\text{cat}=\text{adv}d] \wedge \text{AG} \triangleleft \text{AD}$$

SujetCanon instantiates EpineArg and attaches a noun phrase (SN) substitution node below the argument AD supplied by EpineArg :

$$\text{SujetCanon} \longrightarrow [\text{AD}] \Leftarrow \text{EpineArg}() \wedge \text{SN}_\dagger^R[\text{cat}=\text{sn}, \text{cas}=\text{nom}] \wedge \text{AD} \triangleleft \text{SN}$$

As for the direct and indirect object, they are defined as follows: a direct object is expressed by a noun phrase whose case is accusative. This object normally appears after the verb. If it appears before the verb, an anaphor (referring to the object complement) must be attached at the end of the verb. This requires an object-clitic marker on the verb (boolean feature $\text{oclit}=\text{+}$):

⁵ A black node is a resource and can be unified with 0 or more white nodes; a white node is a need and must be unified with a black node; a red node is saturated and cannot be unified with any other node.

⁶ In our metagrammatical description, tree fragment names are in French (e.g. EpineVerbe) and so are syntactic categories (e.g. SV for Syntagme Verbal).

$$\text{ObjetCanonSN} \longrightarrow [\text{AD}, \text{V}] \Leftarrow \text{EpineArg}() \wedge \text{SN}^R_{\downarrow}[\text{cat}=\text{sn}, \text{cas}=\text{acc}] \\ \wedge \text{AD} \triangleleft \text{SN} \wedge ((\text{V}[\text{oclit}=-] \wedge \text{V} \prec^+ \text{SN}) \vee (\text{V}[\text{oclit}=+] \wedge \text{SN} \prec^+ \text{V}))$$

The direct object can also be just a clitic:

$$\text{ObjetCanonClit} \longrightarrow [\text{V}] \Leftarrow \text{EpineVerbe}(w) \wedge \text{V}[\text{oclit}=+]$$

The indirect object is expressed by a prepositional phrase. It requires a particle P (stipulated by the verb) followed by a noun phrase:

$$\text{ObjetIndCanon} \longrightarrow [\text{AD}, \text{V}] \Leftarrow \text{EpineArg}() \wedge \\ \text{SP}^B[\text{cat}=\text{sp}] \wedge \text{AD} \triangleleft \text{SP} \\ \text{P}^R_{\diamond} \quad \text{SN}^R_{\downarrow}[\text{cat}=\text{sn}, \text{cas}=\text{gen}]$$

Finally, the three basic verb families can be obtained as follows:

$$\text{Intransitive} \longrightarrow \text{MatrixClause} \wedge (\text{SujetCanon} \vee \text{Ellipse})^7 \\ \text{Transitive} \longrightarrow \text{Intransitive} \wedge \text{ObjetCanon}[\text{Objet1}]^8 \\ \text{DiTransitive} \longrightarrow \text{Transitive} \wedge \text{ObjetCanon}[\text{Objet2}]^9$$

3.2 Nominal Sentence

Nominal Sentence in Arabic is a verbless sentence, which consist of a topic followed by a predicate. Following the example of the verbal sentence, we also defined a nominal spine *EpineTheme* instantiated by the *Theme* class:

$$\text{Theme} \longrightarrow \text{EpineTheme}(B)$$

The latter defines the topic of the nominal sentence. The predicate is expressed by and adjective, a prepositional phrase or also a noun phrase:

$$\text{Propos} \longrightarrow [\text{AD}] \Leftarrow \text{EpineArgNom}() \wedge (\text{A}J^R_{\downarrow}[\text{cat}=\text{adj}, \text{cas}=\text{nom}] \vee \\ \text{SP}^R_{\downarrow}[\text{cat}=\text{sp}, \text{cas}=\text{gen}] \vee \text{SN}^R_{\downarrow}[\text{cat}=\text{sn}, \text{cas}=\text{nom}]) \wedge \text{AD} \triangleright \text{SN}$$

By combining these two descriptions, we got several possible combinations of nominal sentences:

$$\text{PhraseNominale} \longrightarrow \text{Theme} \wedge \text{Propos}$$

⁷ For the elliptical subject.

⁸ $\text{ObjetCanon}[\text{Objet1}] \longrightarrow \text{ObjetCanonSN}[\text{Objet1}] \vee \text{ObjetCanonClit}[\text{Objet1}] \vee \text{ObjetIndCanon}[\text{Objet1}]$

⁹ $\text{ObjetCanon}[\text{Objet2}] \longrightarrow \text{ObjetCanonSN}[\text{Objet2}] \vee \text{ObjetCanonClit}[\text{Objet2}] \vee \text{ObjetIndCanon}[\text{Objet2}]$

3.3 Phrasal Structures

Arabic has different types of phrases:

- The noun phrase (مركب اسمي) with has several categories: the annexation phrase (مركب إضافي); the adjectival phrase (مركب نعتي); the corroborative phrase (مركب توكيدي); the approbative phrase (مركب بدلي); the state phrase (مركب بحال المفردة); the conjunctive phrase (مركب العطفي) and the semi-propositional phrase (مركب شبه إسنادي);
- The subordinate phrase (مركب موصولي): it begins with a subordinate conjunction or a relative pronoun and will be followed by a verb.
- The prepositional phrase (مركب حرفي): it consists of a preposition followed by a noun (or a noun phrase).

For each type of phrase, we have defined its corresponding tree description.

4 Dealing with Syntactic Phenomena in ArabTAG V2

4.1 Free Word Order

Arabic has a relatively free word order. Usually nominal sentences begin with a noun or a pronoun, while verbal sentences begin with a verb. The most used order in standard Arabic for a verbal sentence is VSO (V-verb, S-Subject, O-Object). It is possible to change the order of these components without altering the meaning of the sentence. TAG allows combining tree structures without taking into consideration the order of the combinations. Adjunction and/or substitution operations can be called in a free order and can produce sentences with multiple syntactic structures. Moreover, by using XMG's properties, we managed to deal with the semi-free word order within our metagrammar. To do this, we avoided imposing precedence constraints between nodes whose order change does not affect the consistency of the sentence. Let us consider the following sentence: قرأ التلميذ الكتاب (The student read the book). We can change the order of words to have the two combinations قرأ التلميذ قرأ الكتاب (SVO) and قرأ الكتاب التلميذ (VOS). As shown in the Fig. 1¹⁰, our grammar provides all tree models for these three combinations. We used the TuLiPA parser [14] to visualize the derived trees of these sentences.

4.2 The Adjunction of Adverbs

Adjunction in TAG allows the insertion of a complete structure at an interior node of another complete structure. It appears to be a natural way of handling adverbs in natural language. In Arabic, adverbs (adverbial object, circumstantial complement of time, circumstantial complement of place, causative object, etc.)

¹⁰ In order to decrease the size of the image some features have been omitted.

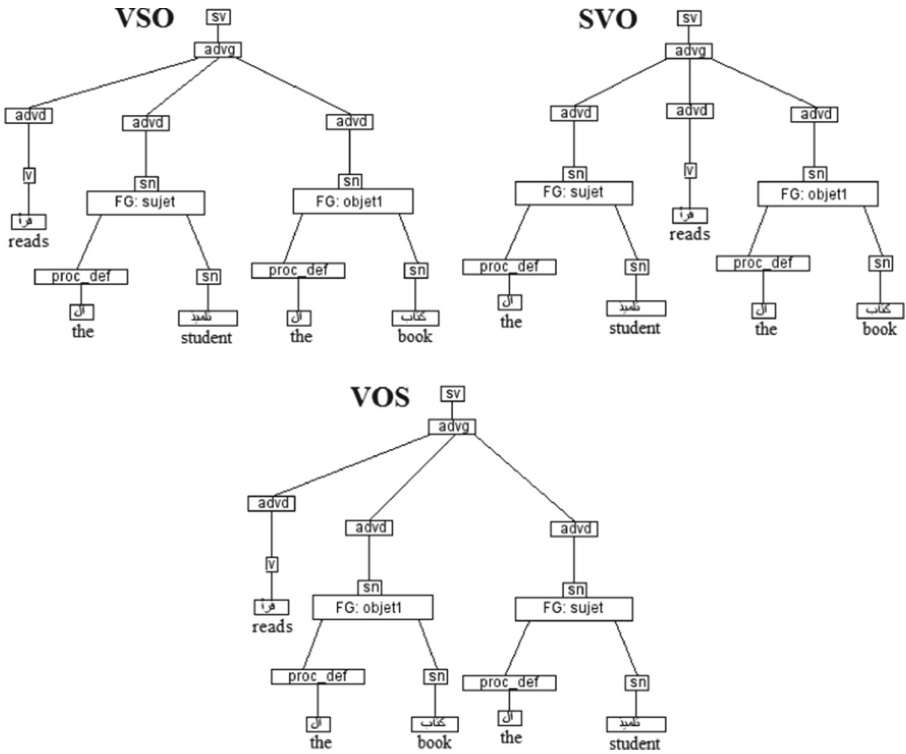


Fig. 1. Free word order of the sentence **قرأ التلميذ الكتاب** (The student read the book)

can be freely interspersed between arguments. We needed to provide two appropriate adjunction points for them: AG (advq) and AD (advd). AG is an adjunction point allowing an adverb at the front of the clause and AD allows inserting an adverb after a verb or an argument. For example, we can add the adverb **كثيراً** (a lot) in the sentence **ينام علي** (Ali sleeps) before the subject **ينام كثيرًا علي** or after the subject **ينام علي كثيرًا** as shown in Fig. 2.

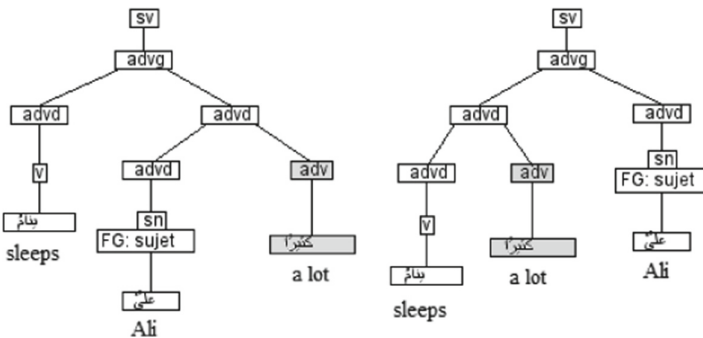


Fig. 2. Adding the adverb **كثيراً** (a lot) in the sentence **ينام علي** (Ali sleeps)

4.3 The Representation of Agglutination Forms

The phenomenon of agglutination consists of joining proclitics and/or enclitics to simple forms of words, which gives rise to more complex forms called agglutinated forms. Proclitic is attached to the beginning of another word (coordinating conjunctions, prepositions, preverbal clitic, etc.). As for enclitic, it is at the end of the word (pronouns, anaphora, etc.). We can also have a sentence consisting of one agglutinated word as in the following example Fig. 3: سيكتبه (He will write it) which is composed of a particle of the future س (will), a verbe يكتب (he writes), and an object ه (it) that are all included in the same text form. To parse an agglutinated form, we must proceed to its division into proclitic/radical/enclitic. This division is itself confronted with a problem of ambiguity since for a single lexical unit we can have several possible divisions. TAG makes it possible to treat this phenomenon thanks to a finite set of possible feature structures associated with the nodes of its elementary trees. These structures contain morphological and syntactic information, which help to assist the parsing procedure and thus remove the ambiguities that may arise. For the example above, we can define in the same feature structure that the proclitic س (will) is a particle of the verb (feature pos: proc_v). This kind of particles can only be attached to the verb in the indicative mode (feature mode: ind). We can notice that the mode of the verb كتب (to write) to which this particle is attached is indeed in the indicative mode يكتب (he writes). Lastly, the enclitic attached to the end of the verb represents its object (feature fg: objet1) with accusative case (feature cas: acc).

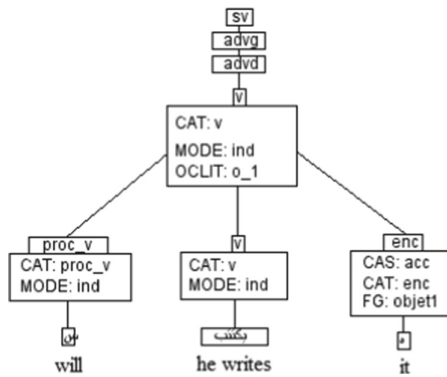


Fig. 3. Derived tree of the sentence سيكتبه (He will write it)

4.4 Subject-Verb and Adjective-Noun Agreement

Agreement rules in Arabic can be summarized as follows:

- Adjectives agree with nouns in definiteness, gender, number, and case
- Subjects and verbs have to agree with each other in both number and gender

- If a verb precedes a plural subject, the verb will always be singular and will agree with the subject only in gender.
- If a plural subject comes before a verb, the verb will agree with the subject in gender and in number.
- If a subject is a non-human plural nouns, the verb should be singular and feminine.

These rules are still followed in our grammar by interesting morphosyntactic feature involved in agreement at the appropriate nodes. These features are Number, gender, personne, case, definiteness and the feature of humanness (humain).

5 Building a Corpus of Phenomenon

In order to verify grammar coverage, we set up a development environment while designing ArabTAG with XMG (see Fig. 4). We defined manually proof of concept syntactic and morphological lexicons for Arabic following the 3-layer lexicon architecture of the XTAG project [13]:

- A basis of tree schemas classified into families of elementary trees
- A lemma basis where each lemma is associated with one (or more) family trees
- A morphological basis in which each flexed form is associated with a lemma and its appropriate morphosyntactic information

The purpose of this validation is to evaluate and to reduce both under and over-generation. Our grammar must be able to distinguish valid sentences that cover linguistic phenomena of Arabic (sentences described in schoolbooks, Arabic novels, etc.) and the ungrammatical sentences. Each new syntactic phenomena included in ArabTAG V2.0 leads to the extension of a test corpus gathering both grammatical and ungrammatical sentences. Every sentence is associated with the number of expected parses. We used the TuLiPA parser [14] on the test corpus to check the quality of the grammar. The parsing results help us to fix

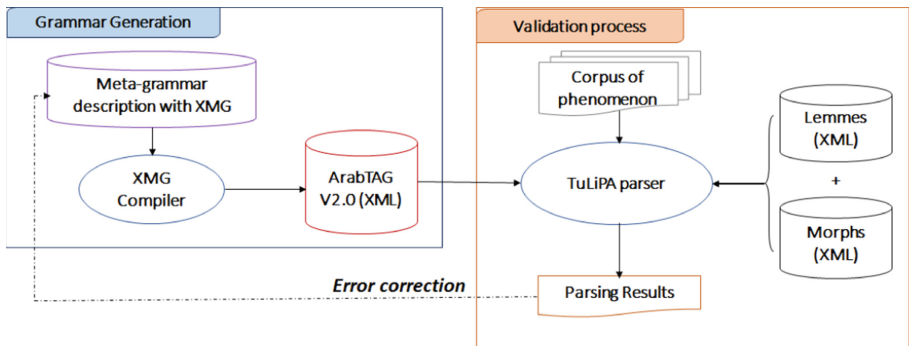


Fig. 4. Validation architecture of ArabTAG V2.0.

potential errors and bugs in our metagrammatical description and allow us to check the consistency of the defined TAG structures when it is extended. By the end of this verification, the test corpus had 217 examples of grammatical and ungrammatical phrases and sentences. Having the initial test corpus developed, we used the parsing result to build a corpus of phenomenon. Correctly parsed sentences¹¹ with its corresponding parse tree (derived tree) are stored into a new corpus called: corpus of phenomenon. The syntactic tree is annotated with morphological information of each lexical entries. The resulted corpus consists of 93 verbal sentences (active and passive form) and 32 nominal sentences. It covers all the phenomena presented in the previous section: : agglutinated form, examples with adverbial object, agreement rules and free word order of syntactic components.

6 ArabTAG V2 Coverage

So far, we have generated 668 trees from a description made of 29 classes (that is, 29 tree fragments or combination rules) as shown in Fig. 5. The current version of the grammar covers verbal phrases (active and passive form), nominal sentences, nominal phrases and prepositional phrases. In addition, it covers elliptical, anaphoric and subordinate structures. It takes into consideration the change of the order of the sentence's components and the agglutinative forms. In addition, it contains elementary trees for the representation of additional complements such as circumstantial complement of time, circumstantial complement of place and adverbs.

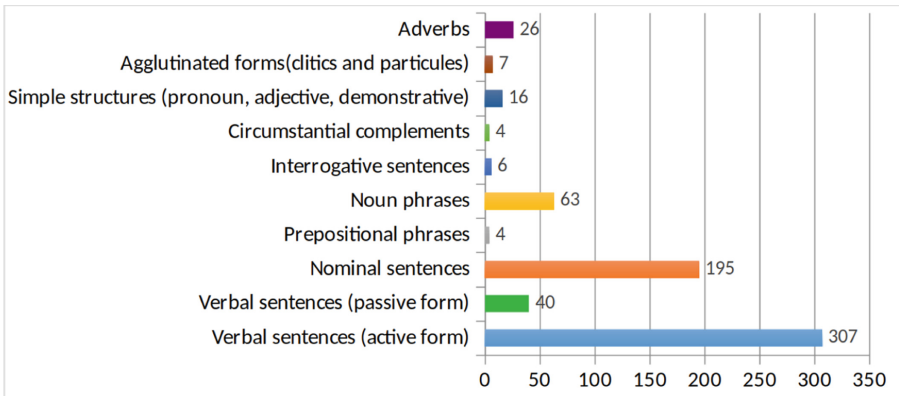


Fig. 5. Current tree distribution in ArabTAG V2.0

¹¹ We did not include phrasal structures.

7 Conclusion

In this paper, we showed how to semi-automatically generate a Tree adjoining grammar representing syntax of Arabic. This grammar is rewritten using the metagrammatical description language XMG. This extensible language offers a mechanism for combining elementary fragments of information and allows information sharing between grammatical structures. In particular, we showed how to describe in a concise and yet easily extensible way, simple verbal subcategorization frames as well as nominal sentence in Arabic. Such a description uses a verbal or noun spine containing adjunction points to deal with the various constituent orders in Arabic. In order to verify grammar coverage, we set up a development environment while designing ArabTAG V2.0. During this process, we used the parsing result of the initial test corpus to build a corpus of phenomenon. The generated grammar covers the simple syntactic structures of the Arabic sentences (verbal and nominal sentences) well as the different phrasal structures (prepositional phrases, noun phrases, etc.). It deals with several linguistic phenomena such as free word order of elements within the syntactic components, the additional complements and the agglutinative forms. In order to integrate semantic information during syntactic analysis, we have extended our meta-grammar by associating semantic frames with the defined families of elementary trees [15]. We decided to use semantic frame as we noted that frame semantic make the interfacing easier between syntax and semantic. This syntax-semantic interface allows the construction of semantic representation of the sentence based on the relationship between its syntactic constituents. Currently, we are evaluating our grammar using a significant syntactic-semantic test corpus. The latter consist of 500 sentences from a Tunisian schoolbook. This choice is due to the unavailability of the resources necessary for such an evaluation.

References

1. Belguith, L., Aloulou, C., Ben Hamadou.: MASPAP: De la segmentation À l'analyse syntaxique de textes arabes. CÉPADUÈS-Editions, editeur, Revue Information Interaction Intelligence I, Vol. 3, 9–36 (2007)
2. Loukam, M., Laskri, M.T.: PHARAS: Une plateforme d'analyse basée sur le formalisme HPSG pour l'arabe standard: Développements récents et perspectives. JED'08, Journées de l'Ecole Doctorale, University Badji Mokhtar, Annaba, Algeria (2008)
3. Attia, M.: Handling Arabic Morphological and Syntactic Ambiguity within the LFG Framework with a View to Machine Translation. Ph.D. Dissertation. University of Manchester, Faculty of Humanities (2008)
4. Habash, N. and Rambow, O.: Extracting a tree adjoining grammar from the penn arabic treebank. In: Proceedings of Traitement Automatique du Langage Naturel (TALN-04), pp. 277–284 (2004)
5. Crabbé, B., Duchier, D., Gardent, C., Le., Roux, J., Parmentier, Y.: XMG?: eXtensible MetaGrammar. *Comput. Linguist.* **39**(3), 591–629 (2013)

6. Ben Khelil, C., Duchier, D., Parmentier, P., Zribi, C., Ben Fraj, F.: ArabTAG : from a Handcrafted to a Semi-automatically Generated TAG, In TAG+12 : 12th International Workshop on Tree-Adjoining Grammars and Related Formalisms, Düsseldorf, Germany (2016)
7. Joshi, A., Levy, L., Takahashi, M.: Tree adjunct grammars. *J. Comput. Syst. Sci.* **10**(1), 136–163 (1975)
8. Maamouri, M., Bies, A., Jin, H., Buckwalter, T.: Arabic treebank: Part 1 v 2.0. LDC Catalog No.: LDC2003T06, ISBN: 1-58563-261-9, ISLRN: pp. 333-321-196-670-5 (2003)
9. Maamouri, M., Bies, A.: Developing an arabic treebank: Methods, guidelines, procedures, and tools. In: Ali Farghaly and Karine Megerdooomian, editors, COLING 2004 Computational Approaches to Arabic Script-based Languages, pp. 2–9, Geneva, Switzerland (2004)
10. Ben Fraj, F.: Construction d’une grammaire d’arbres adjoints pour la langue arabe. In: Actes de la 18e conférence sur le Traitement Automatique des Langues Naturelles, Montpellier, France, June. Association pour le Traitement Automatique des Langues (2011)
11. Kouloughli, D.: La grammaire Arabe pour tous. Press Pocket (1992)
12. Simon Petitjean, S.: Génération Modulaire de Grammaires Formelles. Ph.D. thesis, Université d’Orléans, France (2014)
13. XTAG Research Group.: A lexicalized tree adjoining grammar for english, Technical Report IRCS-01-03, IRCS, University of Pennsylvania (2001)
14. Parmentier, Y., Kallmeyer, L., Lichte, T., Maier, W., Dellert, J.: TuLiPA : A Syntax-Semantics Parsing Environment for Mildly Context-Sensitive Formalisms. In: 9th International Workshop on Tree-Adjoining Grammar and Related Formalisms (TAG+9), 121–128, Tübingen, Germany (2008)
15. Ben Khelil, C., Ben Othmane Zribi, C., Duchier, D., Parmentier, Y.: A new syntactic-semantic interface for ArabTAG an Arabic Tree Adjoining grammar. In: Proceedings of International Arabic Conference of Information Technology (ACIT 2017). Hammamet, Tunisia (2017)



Recognizing Textual Entailment Using Weighted Dependency Relations

Tanik Saikh¹(✉), Sudip Kumar Naskar², and Asif Ekbal¹

¹ Indian Institute of Technology Patna, Bihta, India
{tanik.srf17, asif}@iitp.ac.in

² Jadavpur University, Kolkata, India
sudip.naskar@cse.jdvu.ac.in

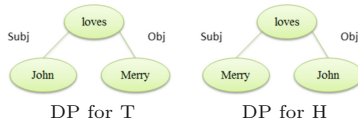
Abstract. In this paper, we describe a hybrid approach for Recognizing Textual Entailment (RTE) that makes use of dependency parsing and semantic similarity measures. Dependency triplet matching is performed between dependency parsed *Text* (T) and *Hypothesis* (H). In case of dependency relation match, we also consider partial matching and semantic similarity between the associated words is calculated with the help of various semantic similarity measures. Importance of various dependency relations with respect to the TE task is computed in terms of their information gain and the dependency relations are weighted accordingly. This paper reports our experiments carried out on the RTE-1, RTE-2 and RTE-3 benchmark datasets using three approaches namely **greedy approach**, **exhaustive search** and **greedy approach with weighted dependency relations**. Experimental results show that weighted dependency relations significantly improve TE performance over the baseline.

Keywords: Dependency parser · Semantic similarity · Textual entailment · Machine learning

1 Introduction

An important feature of natural language is the language variability. There are several ways to express a simple matter. A single piece of text can have various meaning or same meaning can be conveyed by different texts. Textual Entailment (TE) is a kind of problem which would be able to capture such situations. It is one of the toughest problems in natural language processing (NLP) which involves rigorous linguistic analysis and machine learning techniques. It is an unidirectional relation [1] between a pair of texts expressions and it exists if a text called *Text* (T) can be logically inferred from the other one called *Hypothesis* (H). It essentially shows the heredity property between a pair of texts as if H inherits some property from T . It is one of the most prominent research topics in the field of NLP with several important applications such as in *Machine Translation* [2], *Summarization* [3], *Question Answering* [4], *Paraphrase Detection* [5]

and *Novelty/Plagiarism Detection* [6] in a document/text etc and many more. TE between a pair of texts expressions can be determined by observing the lexical, syntactic, semantic information between that particular pair of texts snippets. Literature shows there are many who proposed various studies on it which include lexical [7], syntactic [8], and semantic [9] information. Recently, there has been much interest in applying deep learning models to RTE [10–14]. These models usually do not perform any linguistic analysis. The proposed method can be expressed as the combination of syntactical divergence and semantic similarity which tries to assign weight to each dependency relations produced by dependency parser. This paper proposed an approach to solving the TE problem by taking the help of dependency parsing information and various semantic similarity measures. Dependency parsing of a particular piece of text essentially provides the syntactic representation of that particular piece of text; it produces a number of triplets, each of which essentially represents a relation between two words, the *Governor (G)* and the *Dependent (D)*, like, the triplet "*nsubj (traded-5, delivery-4)*", where *G = traded* and *D = delivery*. Various semantic similarity measures were employed for the purpose namely *Wu-Palmer similarity* [15], *Lin similarity* [16] and *path based similarity* [17]. These metrics are generally used to find semantic similarity between a pair of words, phrases or sentences. Lexical matching suffer from a drawback, occasionally it produced a high score for non-textually entailed texts pair, if we take entailment decision between the text pair cited as follows *T: John loves Merry* and *H: Merry loves John*, by considering n-gram matching, it will produce a high score, consequently, the system will mark that pair as entailed, like the unigram and bigram matching between T and H produces $3/3 = 1$ and $0/3 = 0$ scores respectively. According to unigram matching the sentence pair are textually entails, however they are not.



On the other hand, Dependency Parsing (DP) for the T and H are shown below, where the triplets are *Subj (loves, Merry)* and *Obj (loves, John)*. So we can see none the of *Subj* or *Obj* matches; hence they (text pair) are not entailed, as it is in fact. So dependency matching captures this kind of situations, which is missing in lexical matching. This is the main motivation for using dependency matching in this study.

The motivation behind optimizing dependency relations by assigning weight to them, is the fact that every relation produced by dependency parser for a particular sentence is not equally important, so if we shall be able to assign a weight to each relation and multiply that weight with score obtained from semantic similarity score calculated between two words associated with that particular relation, intuitively, it can be assumed that weighted dependency relation score will increase our system’s performance.

The contributions in this paper can be encapsulated as follows

1. We propound an approach for TE recognition which utilizes dependency parsing information and semantic similarity measures for a T–H pair.
2. We also posit a technique to assign weight to each dependency relation type in order to estimate the importance of each dependency relation with respect to the TE task. We compute the weight of each dependency relation in terms of its normalized information gain.
3. Three sets of experiments namely greedy search, exhaustive search and greedy search with weighted dependency relations are executed.
4. Making use of weighted dependency relations proves to be a very useful technique for recognizing TE.
5. Weighted dependency relations, which we have prepared could be used further for research purpose not only for TE and/or semantic textual similarity but also in any domain of NLP.

1.1 Related Works

Since from a decade, researchers have proposed many different approaches to find the TE relation between a pair of texts. These techniques include the use of dependency parsing of texts, employment of various semantic similarity metrics between a pair of texts.

Literature survey shows several studies on RTE using dependency parsing and WordNet-based semantic similarity were performed. Some of the dependency parsing based TE are reported in [18–29]. These models either exploited parsing information from various parsers and/or semantic information from Wordnet. However, to the best of our knowledge, there is no such study which combines dependency parsing (triplet matching technique, with triplet relation optimization) with semantic similarity to RTE.

The first PASCAL RTE challenge [30] provided a standard platform against which systems can be compared on TE scenario and it was essentially the primary attempt to provide a general task that takes into account major semantic inferences across applications. The challenge received a noticeable response from research communities across the globe; 17 participants took part in it. The participating systems obtained relatively low accuracies which suggest that the task is a challenging one and there are various research avenues in this area. In this challenge, the best result was achieved by [31] using the Bilingual Evaluation Understudy (BLEU) algorithm and obtained an accuracy of 70% on RTE-1 dataset’s Comparable Document (CD) subtask.

A logic-based semantic approach was proposed in [32] which combines the semantic information provided by different resources and extracts new semantic knowledge to improve the system’s performance. The tasks of [33,34] posed machine learning based approaches using conventional similarity metrics (cosine similarity, Jaccard, Dice, etc.), along with machine translation (MT) evaluation metrics (BLEU [35] and METEOR [36] [37]) as features on RTE-1 to RTE-3 datasets and Indian Languages (Tamil, Telugu, Hindi, and Punjabi) respectively. The work defined in [38] made use of three MT evaluation metrics (BLEU,

METEOR, and TER [39]) and one summary evaluation metric (ROUGE [40]) along with polarity (negation) feature on RTE-1, RTE-2, RTE-3, RTE-4, and RTE-5 and obtained a reasonable output as compared to the best performing results in those tracks. The message was there is a correlation between MT evaluation and TE.

The authors in [41] performed several experiments on RTE-3 datasets. They built a system which solely relies on DIRT (Discovering Inference Rules from Text) [42], which is a collection of paraphrases. The system obtained an accuracy of 59.1% on RTE-3 test set. The work of [43] proposed an unsupervised metric to compute the similarity between word pairs in Web1T data. Their proposed approach made use of dependency tree matching technique between text and hypothesis sentences to compute alignment score based on new similarity metric. These scores further used to predict entailment between T-H pairs. The method yielded an overall accuracy of 50.9% on the RTE4 test set.

2 Proposed Approach

T-H pairs are first extracted from the datasets. Parsing of each such T-H pair is performed using the Stanford Dependency parser¹. The parser produces a set of triplets for each sentence. A similarity matrix of order $m * n$ is created from the dependency parse trees of T and H where m and n are the number of triplets in T and H respectively. It was observed in the TE datasets that the texts are typically longer than the hypotheses and therefore $m > n$ usually. A dependency triplet represents a dependency relation between two words in the sentence, the *Governor (G)* and the *Dependent (D)*. We assign weight to each type of dependency relation based on its information gain on the development set. Each dependency triplet of T is compared against each dependency triplet of H. In case of a full match between a T triplet and an H triplet, a score of 1 is assigned to the corresponding T-H triplet pair and insert that value into the appropriate cell in the matrix. Otherwise, the system looks for a matching relation between the T-H triplet pair. If the relation matches, we consider the semantic similarity between the governing words and the semantic similarity between the dependent words of the two triplets. Finally, the average of these two similarity scores is assigned to that T-H dependency triplet pair. The assumption is that the two governing words or the two dependent words in a T-H triplet pair might not be exactly the same words, but they could be synonymous words or related words. In that case, we should consider assigning that triplet pair some non-zero score. Thus instead of considering only binary scores (i.e., 0 or 1) to a triplet pair, we assign scores between 0 and 1. Three semantic similarity metrics, namely *Wu-Palmer (WUP) similarity* [15], *Lin similarity* [16] and *path-based similarity* [17] have been taken into account for this purpose.

Wu-Palmer measures similarity by considering the depth of the two synsets in the WordNet taxonomy, along with the depth of their least common subsumer (LCS). *Lin* measures the similarity between a pair of concepts, say C_1 and

¹ <https://nlp.stanford.edu/software/stanford-dependencies.html>.

C_2 , as $2 * IC(LCS(C_1, C_2))(IC(C_1) + IC(C_2))$, where, $IC(x)$ is the information content of x . *Path-based metric* determines the semantic similarity between two-word senses as an inverse function of the length of the path linking the two concepts. The average of these three semantic similarity metric's scores is taken into consideration as the partial matching score for a T-H triplet pair and is inserted into the appropriate cell in the matrix. Once the triplet matching process terminates, the matrix populated with scores represents the similarity scores between T-H dependency triplets. Assuming that the rows and columns correspond to the dependency relations in H and T, respectively, two or more nonzero scores in a row of this similarity matrix indicates that the corresponding H triplet matches with multiple T triplets. Similarly, two or more nonzero scores in a column in this similarity matrix suggests that the corresponding T triplet matches with multiple H triplets. However, while computing similarity between a T-H dependency tree pair, any H triplet cannot be allowed to match (or align) with multiple T triplets and vice versa. Thus while computing the dependency triplet alignment between a T-H dependency tree pair, we can consider only a maximum of $\min(m, n)$ matching dependency triplets. Therefore, we are interested in finding an optimal combination of p non-zero similarity scores (or cells) from this similarity matrix such that $p < \min(m, n)$, every row and column contribute a maximum of 1 similarity score (or cell) in the combination and the sum of the corresponding similarity scores is highest. Three different approaches were adopted to achieve this objective and to generate the final entailment score for the corresponding T-H sentence pair.

2.1 Greedy Approach

In this approach, first the maximum of all the non-zero values in the matrix is selected and the rest of the non-zero values in the corresponding row and column are set to zero. In every successive iteration, the same process is repeated and iterations continue until there are no more non-zero values left in the matrix. In case of a tie, we select any of the highest scoring cell. Finally, we take the sum of all the selected elements which is further normalized by the number of triplets present in H to arrive at the semantic similarity entailment score for the corresponding T-H pair.

2.2 Exhaustive Search

We exhaustively search over all combinations of p non-zero similarity scores such that $p < \min(m, n)$, and every row and column contribute at most one similarity score in any combination and find out the globally best scoring combination (or alignment). It is to be noted, however, that we are not interested in the combination itself; our objective is to find the optimal similarity score for a T-H dependency tree pair and multiple combinations (or alignments) can yield in the same optimal similarity score. Finally, the optimal sum is normalized by $\min(m, n)$ since a maximum of $\min(m, n)$ similarity scores can contribute to a combination.

2.3 Relation Optimization

Ts and Hs are passed through dependency parser, which yields a collection of relations and words associated with those relations (triplets) for a particular sentence. When comparing T–H dependency triplets for finding TE relation between a pair of texts expressions, all the relations (or triplets) are not equally important and there might be some less important relations which contribute less to the TE recognition process than the other relations. However, the proposed method discussed in the previous section assumes uniform weights for all relations. Therefore, in a bid to improve the overall system performance, we try to assign weight to each dependency relation type according to its importance in TE. We compute the relevance of each dependency relation type in TE in terms of its information gain. For every T–H pair in the dataset, we compute the optimal combination of (fully or partially) matching triplets. It is to be noted however that, in this case, we are interested in the optimal combination and not in the optimal similarity score. Subsequently, we find out how many times each particular dependency relation type contributes to an optimal combination. For every relation, for every T–H pair, we check whether the relation contributes to any optimal triplet combination or not; if it does then a value of 1 is assigned to that relation, otherwise, 0 is assigned. For any relation, the 1 values are considered as $child_1$ and 0 values as $child_0$. We calculate the number of $child_1$ and $child_0$ instances for each relation. To calculate the entropy of $child_1$ and $child_0$ of a relation, we also count how many of its $child_1$ and $child_0$ instances belong to the *TRUE* entailment class and *FALSE* entailment class. Finally, the entropy of its $child_i$ is calculated as in Eq. 1

$$Entropy_{child_i} = -nt/N_i * \log(nt/N_i) - nf/N_i * \log(nf/N_i) \quad (1)$$

where, nt is the number of $child_i$ instances in the TRUE class, nf is the number of $child_i$ instances in the FALSE class, and N_i is the number of $child_i$ instances, i.e., $nt+nf$. The weighted entropy of each child is computed as in Eq. 2.

$$W_E_{child_i} = -Entropy_{child_i} * N_i/N \quad (2)$$

where, nC is the number of instances of $child_i$, and N is the total number of instances of this particular relation in the whole dataset. We used *add-1* smoothing while calculating the entropy to avoid division by zero and/or zero probability. In effect, 1 is added to each of nt and nf , 2 is added to N_i and 4 is added to N while calculating weighted entropy.

Finally, information gain (IG) for the relation is calculated as in Eq. 3.

$$IG_{Relation} = 1 - (W_{E_{child_1}} + W_{E_{child_0}}) \quad (3)$$

We sort the relations in descending order of their information gain values to assess the importance of the relations and normalize the information gains by the sum of all information gains. These normalized information gain values are considered as the weights for the dependency relations.

3 Experiments and Results

This section presents the dataset, experimental setup and the results together with some discussions.

3.1 Dataset

There were many international conferences and evaluation tracks have been organized such as at *Pattern Analysis, Statistical Modeling and Computational Learning (PASCAL)*², *Text Analysis Conferences (TAC)*³ organized by the United States National Institute of Standards and Technology (NIST), Evaluation Exercises on Semantic Evaluation (SemEval)⁴, National Institute of Informatics Test Collection for Information Retrieval System (NTCIR)⁵ since from the year of 2005. These were all dedicated to either recognizing TE between a pair of texts snippets or finding semantic similarity between a pair of texts etc. The experiments were carried out on the datasets released in the PASCAL organized shared task for recognizing textual entailment (RTE) organized in RTE-1, RTE-2 and RTE-3. This paper particularly focusing on two class RTE problem, for that we particularly use RTEs datasets where T–H pairs are defined as two class problem. We would like to port the system to three class problem on Stanford Natural Language Inference (SNLI) Corpus [44] in future. RTE-1 contains 567 and 800 T–H pairs in the development set and test set respectively. Both the development set and test set of RTE-2 and RTE-3 contain 800 entries. Since this work focuses on binary class classification for TE, only RTE-1, RTE-2, and RTE-3 datasets are taken into account. The Table 1 shows true vs false entailment statistics in the dataset.

Table 1. True-False Entailment pair statistics in the dataset

	# of T–H pairs with class	
Dataset	True	False
RTE-1	283	284
RTE-2	400	400
RTE-3	412	288

3.2 Setup

We set various threshold values for taking the entailment decision between T–H pairs and find out the optimum threshold value which maximizes system performance on the development set. The threshold value is obtained by following a

² <http://pascallin.ecs.soton.ac.uk/Challenges/>.

³ <http://www.nist.gov/tac/tracks/index.html>.

⁴ <http://semeval2.fbk.eu/semeval2.php>.

⁵ <http://research.nii.ac.jp/ntcir/ntcir-9/>.

hill-climbing approach where steps are chosen which take us to the highest peak. Following this methodology, if the result obtained with a particular threshold value is better than with another threshold value, we consider our next threshold around the one that yields a better result. Finally, the threshold that provides the best performance on the development set is applied on the corresponding test set. We ran three sets of experiments for each dataset using greedy approach, exhaustive search and greedy search with weighted dependency relations.

3.3 Results

The results obtained with the three approaches on the three datasets are presented in Table 2, where Greedy search, Exhaustive search and greedy search with dependency relations, optimization approaches are represented as Greedy, Exhaustive and RO respectively. A general trend can be observed from the results. The baseline greedy approach produces TE accuracy in the range 55%–56% on all the three datasets. The exhaustive search, as expected, results in some (about 1%–2%) improvements over the baseline greedy approach and produces TE accuracy in the range 56%–58%. The relation optimization approach produces significant improvements (about 5%–8%) over exhaustive search and yields TE accuracy in the range 62%–66%. The improvements provided by an exhaustive search over greedy search and relation optimization approach over exhaustive search are systematic. The relation optimization approach produces the best performance on all three test sets and exhibits 63.12%, 62.37%, and 66.51% accuracies on RTE1, RTE2, and RTE3 respectively. Table 2 also reports the best performances reported in the literature on these three datasets, which are 70% [31], 75% [45] and 80% [46] on RTE1, RTE2 and RTE3 respectively. Thus the proposed approach falls short of the state-of-the-art results obtained on these tracks. However, the proposed approach is a simple unsupervised approach solely relying on matching dependency trees and it does not rely on any other sophisticated tools or techniques, whereas the works reporting state-of-the-art results used many different techniques, e.g., [31] made use of word overlapping

Table 2. Evaluation results on the TE task

Datasets	Approach	Threshold	Accuracy(%)	Best
RTE-1	Greedy	0.4	55.75	70
	Exhaustive	0.0875	56.62	
	RO	0.4625	63.12	
RTE-2	Greedy	0.375	56.25	75
	Exhaustive	0.125	58.56	
	RO	0.25	62.37	
RTE-3	Greedy	0.275	55.75	80
	Exhaustive	0.15	58.25	
	RO	0.55	66.51	

method of the BLEU algorithm [35]. [45] used lexical relations, N-gram subsequence, syntactic matching, semantic role labeling, Web-Based statistics etc., [46] considered discourse commitments, lexical alignment, knowledge extraction from various knowledge Bases, etc. We also compare the results obtain by the proposed system with some existing works exploiting dependency information to TE. Table 3 shows some results of other’s system and the results of the proposed system. The proposed system outperforms the existing system.

Table 3. Comparison with some systems

Group	Dataset	Accuracy(%)
Proposed Approach	RTE-1	63.12
	RTE-2	62.37
	RTE-3	66.51
[41]	RTE-3	59.1
[42]	RTE-4	50.9
[18]	PETE	73.75
[47]	RTE-4	53.86

3.4 Error Analysis

We manually analyzed some of the erroneous cases and the observations are given below.

1. Stanford dependency parser sometimes produces erroneous results, even for short sentences. We also noticed that the Stanford phrase structure parser is excellent compared to the Stanford dependency parser for such cases. In future, we would like to incorporate that into the existing framework to make a comparative study.
2. For many related word pairs (like *Cease and network, ended and went, Police and Police, forms and document, oil and prices, traded and rose* and many more) all the WordNet-based metrics (Wu-Palmer, Lin, path-based similarity) produce semantic similarity score 0, which affects the entailment scores, and in turns affects the entailment decision.
3. It was observed from the dataset that there are many instances of *FALSE* entailment T–H pairs in the training set that yield TE score of 0.5 or higher which makes the threshold, and hence the TE recognition process, difficult to learn. Table 4 presents statistics of such T–H pairs in the datasets.
4. It was also observed that high scoring (≥ 0.5) *FALSE* TE entailment pairs typically contain lots of *Named Entities (NE)* in both T and H. This needs further investigation.

Table 4. Statistics of high scoring (≥ 0.5) *FALSE* TE entailment pairs in the Development sets

Datasets	# of T-H pairs		
	Greedy	Exhaustive	RO
RTE-1	86	76	76
RTE-2	105	72	167
RTE-3	81	58	108

4 Conclusion and Future Work

The paper presents a hybrid approach for TE recognition which exploits dependency parsing information and semantic similarity measures for a T-H text pair. We also tendered a technique which is based on information gain to assign weight to each dependency relation type. We carried out 3 sets of experiments - baseline greedy, exhaustive search and relation optimization, on RTE 1-3. The thresholds were learned from the development sets using a hill-climbing approach. We successfully demonstrated our hypothesis that all the dependency relations are not equally important for the task of TE recognition. Finding the importance of the dependency relations for the TE task through information gain and using them as weights in the T-H dependency tree similarity calculation is the major contribution in the proposed work, which resulted in significant improvements in the TE recognition task.

In future, we would like to apply the proposed system for three class TE problem like what is defined in RTE-4, RTE-5, Stanford Natural Language Inference (SNLI) Corpus [44] and recently released Multi Genre Natural Language Inference corpus (MultiNLI) proposed by [48]. We are also planning to employ Word2Vec model based distributional semantic similarity to remedy the problem of WordNet based semantic similarity.

References

1. Tatar, D., Serban, G., Mihis, A.D., Mihalcea, R., et al.: Textual entailment as a directional relation. *J. Res. Pract. Inf. Technol.* **41**(1), 53 (2009)
2. Hutchins, W.J., Somers, H.L.: *An Introduction to Machine Translation*. Academic Press, London (1992)
3. Radev, D.R., McKeown, K.R.: Generating natural language summaries from multiple on-line sources. *Comput. Linguist.* **24**(3), 470-500 (1998)
4. Green, Jr., B.F., Wolf, A.K., Chomsky, C., Laughery, K.: Baseball: an Automatic Question-Answerer. In: *Papers Presented at the 9-11 May 1961, Western Joint IRE-AIEE-ACM Computer Conference. IRE-AIEE-ACM 1961 (Western)*, New York, USA, ACM, pp. 219-224 (1961)

5. Socher, R., Huang, E.H., Pennington, J., Ng, A.Y., Manning, C.D.: Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In: *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011*. Proceedings of a meeting held 12–14 December 2011, Granada, Spain, pp. 801–809 (2011)
6. Lambert, P., Blanchard, J., Guillet, F., Kuntz, P., Suignard, P.: Novelty detection in text streams - a survey. In: *European Conference on Data Analysis. Proceedings of the European Conference on Data Analysis, Luxembourg, Luxembourg (2013)*
7. Dagan, I., Glickman, O., Magnini, B.: The PASCAL recognising textual entailment challenge. In: Quiñonero-Candela, J., Dagan, I., Magnini, B., d’Alché-Buc, F. (eds.) *MLCW 2005. LNCS (LNAI)*, vol. 3944, pp. 177–190. Springer, Heidelberg (2006). https://doi.org/10.1007/11736790_9
8. Vanderwende, L., Dolan, W.B.: What syntax can contribute in the entailment task. In: Quiñonero-Candela, J., Dagan, I., Magnini, B., d’Alché-Buc, F. (eds.) *MLCW 2005. LNCS (LNAI)*, vol. 3944, pp. 205–216. Springer, Heidelberg (2006). https://doi.org/10.1007/11736790_11
9. Burchardt, A., Reiter, N., Thater, S., Frank, A.: A semantic approach to textual entailment: system evaluation and task analysis. In: *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing. RTE 2007*, pp. 10–15. Stroudsburg, PA, USA, Association for Computational Linguistics (2007)
10. Bowman, S.R., Angeli, G., Potts, C., Manning, C.D.: Learning natural language inference from a large annotated corpus. In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 632–642. Stroudsburg, PA, Association for Computational Linguistics (2015)
11. Rocktäschel, T., Grefenstette, E., Hermann, K.M., Kocisky, T., Blunsom, P.: Reasoning about entailment with neural attention. In: *International Conference on Learning Representations (ICLR) (2016)*
12. Wang, S., Jiang, J.: Learning natural language inference with LSTM. In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego, California, Association for Computational Linguistics*, pp. 1442–1451 (June 2016)
13. Parikh, A., Täckström, O., Das, D., Uszkoreit, J.: A decomposable attention model for natural language inference. In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Austin, Texas, Association for Computational Linguistics*, pp. 2249–2255 (November 2016)
14. Sha, L., Chang, B., Sui, Z., Li, S.: Reading and thinking: re-read LSTM unit for textual entailment recognition. In: *COLING. (2016)*
15. Wu, Z., Palmer, M.: Verbs Semantics and Lexical Selection. In: *Proceedings of the 32Nd Annual Meeting on Association for Computational Linguistics. ACL 1994*, pp. 133–138. Stroudsburg, PA, USA, Association for Computational Linguistics (1994)
16. Lin, D.: An Information-Theoretic Definition of Similarity. In: *Proceedings of the Fifteenth International Conference on Machine Learning. ICML 1998*, pp. 296–304. San Francisco, CA, USA, Morgan Kaufmann Publishers Inc. (1998)
17. Pedersen, T., Patwardhan, S., Michelizzi, J.: WordNet: :Similarity: measuring the relatedness of concepts. In: *Demonstration Papers at HLT-NAACL 2004. HLT-NAACL-Demonstrations 2004*, pp. 38–41. Stroudsburg, PA, USA, Association for Computational Linguistics (2004)
18. Basak, R., Naskar, S.K., Pakray, P., Gelbukh, A.F.: Recognizing Textual Entailment by Soft Dependency Tree Matching. *Computación y Sistemas* **19**(4), 685–700 (2015)

19. Herrera, J., Peñas, A., Verdejo, F.: Textual entailment recognition based on dependency analysis and WordNet. In: Quiñonero-Candela, J., Dagan, I., Magnini, B., d'Alché-Buc, F. (eds.) *MLCW 2005*. LNCS (LNAI), vol. 3944, pp. 231–239. Springer, Heidelberg (2006). https://doi.org/10.1007/11736790_13
20. Snow, R., Vanderwende, L., Menezes, A.: Effectively using syntax for recognizing false entailment. In: *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference, New York, USA, Association for Computational Linguistics*, pp. 33–40 (June 2006)
21. Pakray, P., Bandyopadhyay, S., Gelbukh, A.: Dependency parser based textual entailment system. In: *Artificial Intelligence and Computational Intelligence, International Conference on (AICI)*. Vol. 01, pp. 393–397 (Nov 2010)
22. Rus, V., G.A.M.P.L.K.: A study on textual entailment. In: *Proc. of 17th International Conference on Tools with Artificial Intelligence (ICTAI 2005)*, pp. 326–333 IEEE. (2005)
23. Marsi, E., Krahmer, E., Bosma, W., Theune, M.: Normalized alignment of dependency trees for detecting textual entailment. In: Springer Verlag, pp. 56–61 (Apr 2006)
24. Kouylekov, M., Magnini, B.: Recognizing textual entailment with tree edit distance algorithms. In: *PASCAL Challenges on RTE*, pp. 17–20 (2005)
25. Haghighi, A.D., Ng, A.Y., Manning, C.D.: Robust textual inference via graph matching. In: *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing, HLT 2005, Stroudsburg, PA, USA, Association for Computational Linguistics*, pp. 387–394 (2005)
26. Sidorov, G.: Should syntactic N-grams contain names of syntactic relations? *Int. J. Comput. Linguistics Appl.* **5**(2), 25–47 (2014)
27. Dash, S.K., Pakray, P., Gelbukh, A.F.: Learning physiotherapy through virtual action. *Computación y Sistemas* **20**(3), 477–482 (2016)
28. Alabbas, M., Ramsay, A.: Dependency tree matching with extended tree edit distance with subtrees for textual entailment. In: *Federated Conference on Computer Science and Information Systems - FedCSIS 2012, Wroclaw, Poland, 9–12 September 2012, Proceedings*, pp. 11–18 (2012)
29. Marsi, E., Krahmer, E., Bosma, W., Theune, M.: Normalized alignment of dependency trees for detecting textual entailment. In: Magnini, B., Dagan, I., (eds.): *Second PASCAL Recognising Textual Entailment Challenge*, pp. 56–61 (April 2006)
30. Dagan, I., Glickman, O.: Probabilistic textual entailment: generic applied modeling of language variability. In: *Learning Methods for Text Understanding and Mining*. (Jan 2004)
31. Perez, D., Alfonsecaia, E., Rodríguez, P.: Application of the Bleu Algorithm for recognising textual entailments. In: *Proceedings of the Recognising Textual Entailment Pascal Challenge*. (2005)
32. Tatu, M., Moldovan, D.: A Semantic Approach to Recognizing Textual Entailment. In: *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing, HLT 2005, Stroudsburg, PA, USA, Association for Computational Linguistics*, pp. 371–378 (2005)
33. Saikh, T., Naskar, S.K., Giri, C., Bandyopadhyay, S.: Textual entailment using different similarity metrics. In: Gelbukh, A. (ed.) *CICLing 2015*. LNCS, vol. 9041, pp. 491–501. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-18111-0_37
34. Saikh, T., Naskar, S.K., Bandyopadhyay, S.: JU_NLP@DPIL-FIRE2016: Paraphrase Detection in Indian Languages - A Machine Learning Approach. In: *Working notes of FIRE 2016 - Forum for Information Retrieval Evaluation, Kolkata, India, December 7–10, 2016*, pp. 275–278 (2016)

35. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: BLEU: a method for automatic evaluation of machine translation. In: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics. ACL 2002, pp. 311–318 Stroudsburg, PA, USA, Association for Computational Linguistics (2002)
36. Banerjee, S., Lavie, A.: METEOR: an automatic metric for MT evaluation with improved correlation with human judgments. In: Proceedings of Workshop on Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization at the 43rd Annual Meeting of the Association of Computational Linguistics (ACL-2005), Ann Arbor, Michigan, June 2005. (2005)
37. Lavie, A., Agarwal, A.: METEOR: An Automatic Metric for MT Evaluation with High Levels of Correlation with Human Judgments. In: Proceedings of the Second Workshop on Statistical Machine Translation. StatMT 2007, Stroudsburg, PA, USA, Association for Computational Linguistics, pp. 228–231 (2007)
38. Saikh, T., Naskar, S., Ekbal, A., Bandyopadhyay, S.: Textual entailment using machine translation evaluation metrics. In: Computational Linguistics and Intelligent Text Processing - 18th International Conference on Computational Linguistics and Intelligent Text Processing, pp. 17–23. CICLing 2017, Budapest, Hungary (April 2017)
39. Snover, M., Dorr, B., Schwartz, R., Micciulla, L., Makhoul, J.: A study of translation edit rate with targeted human annotation. In: Proceedings of Association for Machine Translation in the Americas, pp. 223–231 (2006)
40. Lin, C.Y.: ROUGE: a package for automatic evaluation of summaries. In: Proceedings of the Workshop on Text Summarization Branches Out (WAS 2004), Barcelona, Spain, 25–26 July 2004. (2004)
41. Marsi, E., Kraher, E., Bosma, W.: Dependency-based paraphrasing for recognizing Textual Entailment. In: Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing. RTE 2007, Stroudsburg, PA, USA, Association for Computational Linguistics, pp. 83–88 (2007)
42. Lin, D., Pantel, P.: DIRT @SBT@Discovery of Inference Rules from Text. In: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD 2001, New York, USA, pp. 323–328 ACM (2001)
43. Yatbaz, M.A.: RTE4: Normalized dependency tree alignment using unsupervised N-gram word similarity score. In: Proceedings of the First Text Analysis Conference, TAC 2008, Gaithersburg, Maryland, USA, 17–19 November 2008, Gaithersburg, Maryland, USA, NIST (2008)
44. Bowman, S.R., Angeli, G., Potts, C., Manning, C.D.: A Large Annotated Corpus for Learning Natural Language Inference. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17–21, 2015, pp. 632–642 (2015)
45. Hickl, A., Bensley, J., Williams, J., Roberts, K., Rink, B., Shi, Y.: Recognizing Textual Entailment with LCCs Groundhog System. (2005)
46. Hickl, A., Bensley, J.: A Discourse Commitment-based Framework for Recognizing Textual Entailment. In: Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing. RTE 2007, Stroudsburg, PA, USA, Association for Computational Linguistics pp.171–176 (2007)
47. Pakray, P., Gelbukh, A., Bandyopadhyay, S.: A syntactic textual entailment system based on dependency parser. In: Gelbukh, A. (ed.) CICLing 2010. LNCS, vol. 6008, pp. 269–278. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-12116-6_22
48. Williams, A., Nangia, N., Bowman, S.R.: A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. CoRR abs/1704.05426 (2017)



Hypernymy Detection for Vietnamese Using Dynamic Weighting Neural Network

Van-Tan Bui^{1(✉)}, Phuong-Thai Nguyen², and Van-Lam Pham³

¹ University of Economic for Technical Industries, Hanoi, Vietnam
bvtan@uneti.edu.vn

² University of Engineering and Technology, Vietnam National University, Hanoi, Vietnam
thainp@vnu.edu.vn

³ Institute of Linguistics, Vietnam Academy of Social Sciences, Hanoi, Vietnam

Abstract. The hypernymy detection problem aims to identify the “is-a” relation between words. This problem has recently received attention from researchers in the field of natural language processing because of its application to varied downstream tasks. So far, fairly effective methods for hypernymy detection in English have been reported. In Vietnamese, this problem has not been effectively solved. In this study, we applied a number of hypernymy detection methods based on word embeddings and supervised learning for Vietnamese. We propose an improvement on the dynamic weighting neural network model introduced by Luu Anh Tuan et al. [18] by weighting context words proportionally to the semantic similarity between them and the hypernym. Based on Vietnamese WordNet, three datasets for hypernymy detection were built. Experimental results showed that our proposal can increase the efficiency from 8% to 10% in terms of accuracy compared to the original method.

Keywords: Hypernymy detection · Taxonomic relation · Lexical entailment

1 Introduction

Hypernymy is the relationship between a generic word (hypernym) and its specific instance (hyponym). For example, *vehicle* is a hypernym of *car* while *fruit* is a hypernym of *mango*. This relationship has recently been studied extensively from different perspectives in order to develop the mental lexicon [22]. In addition, hypernymy is also referred to as the taxonomic [18], is-a [24], or inclusion relations [22]. Hypernymy is one of the most basic relations in many structured knowledge databases such as WordNet [8] and BabelNet [20].

There are a number of important characteristics of the Vietnamese language that impact the hypernymy detection problem. Firstly, Vietnamese is an isolating language in which words do not change their forms according to their

grammatical function in a sentence. Secondly, the smallest unit in the formation of Vietnamese words is the syllable. Words can have just one syllable such as *đẹp* <beautiful>, *đắt* <expensive>, or be a compound of two or more syllables such as *màu sắc*¹<color>. Thirdly, as in many other Asian languages such as Chinese, Japanese and Thai, there is no word delimiter in Vietnamese. The space is a syllable delimiter but not a word delimiter, so a Vietnamese sentence can often be segmented in many ways [22].

The automatic hypernymy detection has been applied effectively in many NLP tasks such as taxonomy creation [21,25], recognizing textual entailment [7], and text generation [5]. Among many others, a good example is presented in [28] about recognizing entailment between sentences by identifying hypernymy relation between words. For example, since bitten is a hyponym of attacked, and dog is a hyponym of animal, “*George was bitten by a dog*” and “*George was attacked by an animal*” have an entailment relation.

Previous studies on this problem can be categorized into two main approaches including statistical learning and linguistic pattern matching [18]. The linguistic approach relies on lexical-syntactic patterns capturing textual expressions of taxonomic relations to identify the hypernymy relation between pairs of words in a corpus. For example, Hearst presented a pioneer work to extract is-a relations from a text corpus based on handcraft patterns [11]. The following up works mostly focus on is-a relation extraction using automatically generated patterns [14,25].

Following the statistical learning approach, several studies are based on distributional representation [4,13,23,29]. Word embeddings such as GloVe and Word2Vec have shown promise in a variety of NLP tasks including hypernymy detection. Word representations are constructed to minimize the distance between words with similar contexts. According to the distributional similarity hypothesis [10], this means that similar words should have similar representations. However, these methods make no guarantees about more fine grained semantic properties [12].

In recent years, word embeddings have been exploited in conjunction with supervised learning to detect relations between word pairs. Omer Levy et al. [16] pointed out that using linear SVMs, as the foregoing work has done, reduces the classification task to that of predicting whether in a pair of words, the second one has some general properties associated with being a hypernym [16]. Several studies on hypernymy relation detection using word embeddings (i.e. Word2Vec and GloVe) [9,27].

In recent years, word embeddings have been exploited in conjunction with supervised learning to detect relations between word pairs. Omer Levy et al. [16] pointed out that using linear SVMs, as the foregoing work has done, reduces the classification task to that of predicting in a word pair whether the second one has some general properties associated with being a hypernym [16]. Several studies on hypernymy relation detection using word embeddings (i.e. Word2Vec and GloVe) [9,27].

¹ In this paper, we used ‘_’ characters to associate the syllables of a compound word in Vietnamese.

In this paper, we present an idea to improve the method proposed by Luu et al. [18]. Our idea is that context words should not be weighted uniformly. We assume that the role of context words is uneven. The more similar a context word is to the hypernym, the higher weight the context word is assigned. We propose a specific method for weighting context words. We then apply the new embedding as features for hypernymy detection using a support vector machine model. Since hypernymy detection for Vietnamese is a new problem, there is no dataset published yet. Based on Vietnamese WordNet and a large corpus of Vietnamese texts, we built three datasets for hypernymy detection. Experimental results demonstrated that our proposal can increase the performance compared to the original method.

The rest of this paper is structured as follows. Section 2 presents our improvement proposal on the word embedding model. Section 3 describes the construction of hypernymy datasets for Vietnamese. Section 4 presents experimental results and evaluation. The last section gives conclusions.

2 Methodology

According to the DWN method [18], the role of context words is the same in a training sample, each word is assigned a coefficient $\frac{1}{k}$, whereas hyponym has the coefficient k to reduce the bias problem of a high number of contextual words. By observing the triplets extracted from the Vietnamese corpus, we can see that some of them have a high number of contextual words, the semantic similarity between each contextual word and the hypernym is different, as shown in Table 1. We assume that *the role of contextual words is uneven, words that have higher semantic similarity with hypernym should be assigned a greater weight*. Therefore, we suppose that the weight for contextual words is proportional to the semantic similarity between them and hypernym. Through this weighting method, it is possible to reduce the bias of many contextual words that they themselves are less important.

2.1 Learning Word Embeddings

In recent years, word embeddings have shown promise in a variety of NLP tasks. The most typical of these techniques is Word2Vec [19], with two models Skip-gram and Continuous bag of words (CBOW). The CBOW model is roughly the mirror image of the Skip-gram model. It is based on a predictive model predicting the current word from the context window of $2n$ words around it (Eq. 1).

$$O = \frac{1}{T} \sum_{t=1}^T \log(P(w_t | w_{t-n}, \dots, w_{t-1}, w_{t+1}, \dots, w_{t+n})) \quad (1)$$

Training steps are similar to the DWN model. There are three steps for learning word embeddings: firstly, extracting hypernymy pairs from Vietnamese WordNet; secondly, extracting training triplets from corpus; finally, training a neural network, in this step, for each of the triplets in the training set, we complement semantic similarity coefficient between contextual words and the hypernym.

Table 1. Several triplets.

Sentence	Hypernym–Hyponym	Contextual words
<i>Một trong những loài hoa có gai nhọn, có nhiều màu_sắc và hương_thơm quyến_rũ là hoa_hồng</i> <One of the flowers that have sharp thorns, many colors and seductive fragrances is rose>	hoa<flower> hoa_hồng<rose>	- có_gai_nhọn, nhiều_màu_sắc và_hương_thơm quyến_rũ là
<i>voi là loài ăn thực_vật nên chúng thường sống ở khu_vực rừng nhiệt_đới có nhiều cỏ, chúng là loài động_vật sống trên cạn to_lớn nhất còn tồn_tại cho đến ngày_nay</i> <elephants are herbivores so they live in tropical forests where there is a lot of grass, they are the largest terrestrial animals that have been alive until now>	động_vật<animal> voi<elephant>	- là loài ăn thực_vật nên chúng thường sống ở khu_vực rừng nhiệt_đới có nhiều cỏ, chúng là loài

Vietnamese WordNet: Princeton WordNet is a large lexical database for the English language [8]. Currently, Vietnamese WordNet (see Fig. 1) has been constructed based on the Princeton WordNet and applied quite effectively in studies on Vietnamese natural language processing [26]. Vietnamese WordNet contains 32,413 synsets, 66,892 words [22].

Semantic Similarity Measurement: To evaluate the semantic similarity level between contextual words and hypernym, we use the Lesk algorithm [15] which was proposed by Michael E. Lesk for word sense disambiguation problem can measure the similarity based on the gloss of words, with the hypothesis two words are similar if their definitions share common words [2]. This algorithm is used because of the following reasons. Firstly, it only uses the brief definition of words in the dictionary instead of using the structural information of Vietnamese WordNet. Second, its performance is better than other knowledge-based methods. Furthermore, a study has shown that this algorithm gives the best results for the semantic similarity problem in Vietnamese [26]. The similarity of a word pair is defined as a function that overlaps the corresponding definitions (glosses) provided by a dictionary (Eq. 2):

$$Sim_{Lesk}(w_1, w_2) = Overlap(gloss(w_1), gloss(w_2)) \tag{2}$$

In Vietnamese WordNet, *vợ* <wife>, *chồng* <husband> are defined as follows: *vợ*: “*người phụ_nữ đã kết_hôn, trong quan hệ với người đàn_ông kết_hôn với mình*”<a married woman; a man’s partner in marriage>
chồng: “*người đàn_ông đã kết_hôn, hôn phu của người phụ_nữ trong hôn nhân*”<a married man; a woman’s partner in marriage>

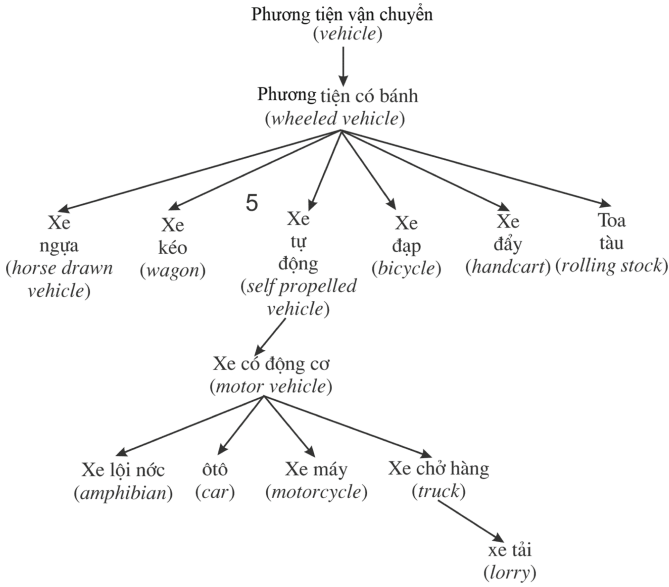


Fig. 1. A fragment of the Vietnamese WordNet hypernym hierarchy.

To achieve the good word similarity values by the Lesk algorithm, we used the extended gloss idea which was presented in [1], and then applied it in Vietnamese [26].

Extracting Data: The purpose of this step is to extract a set of hypernymy pairs for training from Vietnamese WordNet. The total number of hypernymy pairs is 269,781. After that, we extract the triplets of hypernym, hyponym, and the set of context words between them. Context words are all words located between hypernym and hyponym in a sentence. Using the set of hypernymy pairs extracted from the first step as a reference, we extract from the corpus all sentences which contain at least two words involved in this list. Corpus used in this study contains about 21 million sentences (about 560 million tokens), which are crawled from the internet and then filtered, standardized, and segmented. In total, we have extracted 2,985,618 training triplets from this corpus, this list contains 138,062 hypernymy pairs.

In a triplet $\langle hype, hypo, contextual\ words \rangle$, with each contextual word x_{ct} , we define the coefficient α_t which is proportional to the semantic similarity between x_{ct} and hypernym. The word similarity is evaluated by the Lesk algorithm based on their glosses in Vietnamese WordNet, α_t defined in Eq. 3.

$$\alpha_t = \frac{Sim_{Lesk}(x_{ct}, hype)}{\sum_{i=1}^k Sim_{Lesk}(x_{ci}, hype)} \tag{3}$$

Note that $\sum_{i=1}^k \alpha_i = 1$, where k is the number of contextual words.

2.2 Proposed Model

To evaluate the semantic similarity between contextual words and hypernym, we use the Lesk algorithm [15] which was proposed by Michael E. Lesk for the word sense disambiguation problem. This algorithm can measure the semantic similarity between words based on their glosses, with the hypothesis *two words are similar if their definitions share common words*. The Lesk algorithm is used because of the following reasons. Firstly, it only uses the brief definition of words in the dictionary instead of using the structural information of Vietnamese WordNet. Second, its performance is better than other knowledge-based methods. Furthermore, a study has shown that this algorithm gives the best results for the semantic similarity problem in Vietnamese [26]. The similarity of a pair of words is defined as a function that overlaps the corresponding definitions (glosses) that are provided by the dictionary (Eq. 4).

$$Sim_{Lesk}(w_1, w_2) = overlap(gloss(w_1), gloss(w_2)) \tag{4}$$

In a triplet $\langle hype, hypo, contextual\ words \rangle$, with each contextual word x_{ct} , we define a coefficient α_t is proportional to the semantic similarity between x_{ct} and $hype$ ($\sum_1^k \alpha_i = 1$, where k is the number of contextual words).

$$\alpha_t = \frac{Sim_{Lesk}(x_{ct}, hype)}{\sum_1^k Sim_{Lesk}(x_{ci}, hype)} \tag{5}$$

Denote $x_{contexts}$ as the summation vector of the context vectors, k -context word in each triplet is calculated as follows:

$$x_{contexts} = \sum_1^k \alpha_i x_{c_i} \tag{6}$$

Let v_t is denoted the vector representation of the input word t , v_t and $v_{contexts}$ as follows:

$$v_t = x_t^T W \tag{7}$$

$$v_{contexts} = x_{contexts}^T W$$

The output of hidden layer h is calculated as:

$$h = \frac{v_{hypo} + v_{contexts}}{2} \tag{8}$$

From the hidden layer to the output layer, there is a different weight matrix $W'_{N \times V}$. Each column of W' is a n -dimensional vector v'_t represents the output vector of word t . Using these weights, we can compute a score u_t for each word in the vocabulary:

$$u_t = v'_t{}^T . h \tag{9}$$

We use the Softmax function as a log-linear classification model to obtain the posterior distribution of hypernym word. In another word, it is a multinomial distribution (Eq. 10).

$$p(\text{hype}|\text{hypo}, c_1, c_2, \dots, c_k) = \frac{e^{u_{\text{hype}}}}{\sum_1^V e^{u_i}} = \frac{e^{v_{\text{hype}}^T \times \frac{v_{\text{hypo}} + v_{\text{contexts}}}{2}}}{\sum_1^V e^{v_i^T \times \frac{v_{\text{hypo}} + v_{\text{contexts}}}{2}}} \quad (10)$$

Then objective function is defined as:

$$O = \frac{1}{T} \sum_{t=1}^T \text{Log}(p(\text{hype}_t|\text{hypo}_t, c_{1t}, c_{2t}, \dots, c_{kt})) \quad (11)$$

Herein, $t = \langle \text{hype}_t, \text{hypo}_t, c_{1t}, c_{2t}, \dots, c_{kt} \rangle$ is a sample in training data set T , hype_t , hypo_t , c_{1t} , c_{2t}, \dots, c_{kt} respectively hypernym, hyponym and contextual words. After maximizing the log-likelihood objective function in Eq. 11 over the entire training set using stochastic gradient descent, the word embeddings are learned accordingly.

Both Word2vec and the proposed model are prediction models, in which word2vec model relies on the *distributional hypothesis* (Harris, 1954; Firth, 1957), *in which words with similar distributions (shared context) have related meaning (have the same vector)*. Word2vec predicts contextually when has a target word on each training sample (Skip-Gram), or vice versa (CBOW). Different from Word2vec, the proposed model predicts a hypernym when has a hyponym and a context on each triplet in the training corpus. *According to this objective training, achieved vectors to obey a hypothesis, in which words have similar hyponyms and share contexts to have near vectors.*

2.3 Supervised Hypernymy Detection

Recently, a number of studies use support vector machine (SVM) [6] for relation detection especially for LE recognition [16]. In this work, SVM is also used to identify pair of words represented by embeddings vectors are LE relation or not. Linear SVM is used because of its speed and simplicity. We used the *ScikitLearn*² implementations with default settings. We create a unique feature vector for the SVM's input from two distributional vectors of words. Inspired by the experiments of [29], several combinations of vectors are experimented and reported (Table 2).

² <http://scikit-learn.org>.

Table 2. Several combinations of vectors

V_{DIFF}	The vector difference ($v_{hyype} - v_{hyypo}$)
V_{MULT}	The pointwise product vector ($v_{hyype} * v_{hyypo}$)
V_{ADD}	The vector sum ($v_{hyype} + v_{hyypo}$)
V_{CAT}	The vector concatenation ($v_{hyype} \oplus v_{hyypo}$)
V_{CATs}	The concatenation vector of sum and difference vector ($\langle v_{hyype} + v_{hyypo} \rangle \oplus \langle v_{hyype} - v_{hyypo} \rangle$)

3 A New Vietnamese Hypernymy Dataset

Datasets play an important role in the field of relation detection problems. The construction of accurate and valid datasets is a challenge [4, 29]. So far, standard datasets for this problem in Vietnamese have not been published yet. For the purpose of constructing Vietnamese datasets, we review several datasets³ that have been published for English, these datasets have been used for experiments in [16] (Table 3).

Table 3. Several datasets.

Dataset	#Instances	#Positive	#Negative
BLESS	14,547	1,337	13,210
ENTAILMENT	2,770	1,385	1,385
Turney 2014	1,692	920	772
Levy 2014	12,602	945	11,657

BLESS dataset: BLESS is a collection of examples of hypernyms, co-hyponyms, meronyms, and random unrelated words for each of 200 concrete, largely monosemous nouns [4]. ENTAILMENT dataset: It consists of 2,770 pairs of terms, with an equal number of positive and negative examples of hypernymy relation. Altogether, there are 1,376 unique hyponyms and 1,016 unique hypernyms [3]. Turney and Mohammad dataset: is based on a crowdsourced dataset of 79 semantic relations. Each semantic relation was linguistically annotated as entailing or not [28]. Levy dataset: is based on manually annotated entailment graphs of subject-verb-object tuples. This is the most realistic dataset since the original entailment annotations were made in the context of a complete proposition [16].

Analyze the differences between hypernymy in English and Vietnamese, based on the structure of published datasets for English, especially the criteria given by Julie Weeds et al. [29] for benchmark datasets, the requirements for a Vietnamese dataset are as follows:

- The dataset should contain words that belong to different domains.

³ <https://github.com/ahug/HypEval/tree/master/data>.

- A dataset needs to be balanced in many respects in order to prevent the supervised classifiers from making use of artifacts of the data.
- There should be an equal number of positive and negative examples of semantic relation.
- The negative examples need to be pairs of equally similar words, but where the relationship under consideration does not hold.
- The number of words in the dataset, should balance in classes (e.g. city, actor, ...) and instances (e.g. Paris, Tom Cruise, ...).

To visualize the structure of Vds1, Vds2, and Vds3 datasets⁴, they are represented as graphs structure. Vertices represent words, edges represent hypernymy relation (see Fig. 2, 3).

Vds1 dataset: The words of this dataset are selected from Vietnamese WordNet and they belong to different domains: plants, animals, furniture, foods, materials, vehicles, and others. Each pair of words in the dataset is assigned one of the three semantic relation labels.

- Hypernym: is hypernym of, (e.g. hoa <flower> - hoa_hồng <rose>).
- Co-hyponym: that is a co-hyponym (coordinate) of, (e.g. hoa_hồng <rose>- hoa_hướng_dương <sunflower>).
- Random: has no hypernym or co-hyponym relation with, (e.g. hoa <flower> - xe_đạp <bicycle>).

Vds2 dataset: This dataset consists of 1,657 hypernymy pairs which are chosen from 269,781 hypernymy pairs extracted from Vietnamese WordNet (Table 4). Figure 2a shows that the Vds1 dataset contains hypernymy pairs and they belong to some domains, some words share a hypernym forming tree structure. In contrast, Fig. 2b shows that most of the hypernymy pairs are disjoint pairs because they are randomly selected from Vietnamese WordNet.



Fig. 2. Visualization of the datasets.

Vds3 dataset: We extracted from Vietnamese WordNet two subnets. The first subnet contains hypernymy pairs extracted from the taxonomy tree, which is a subtree with the root node as động_vật <animal> (Vds3_{animal}); The second subnet is a subtree with the root node as thực_vật <plant> (Vds3_{plant}). In other

⁴ <https://github.com/BuiTan/Vds>.

words, these subnets are taxonomy trees. The height of the tree which corresponds to $Vds3_{animal}$ is 12 and contains 2,284 hypernymy pairs. For $Vds3_{animal}$, the height of the tree is 9 and contains 2,267 hypernymy pairs. Figure 3 visualizes two subnets, in which Fig. 3a shows $Vds3_{animal}$, and Fig. 3 shows $Vds3_{plant}$. The number of pairs for each relation from the three datasets is summarized in Table 4.

4 Evaluation

We conduct experiments to evaluate the performance of the improved method compared to other methods. Three techniques of word embeddings are implemented: Word2Vec4 model [19], DWN [18], and our improved DWN model (our). Training the Word2Vec model in Vietnamese, we use a corpus that contains about 21 million sentences (about 560 million words), we exclude from this corpus any word that appears less than 50 times. Data for training DWN and improved DWN model has 2,985,618 triplets and 138,062 individual hypernymy pairs which are extracted from the above corpus. To decide whether a word pair hold the hypernymy relation, we build a classifier that uses embedding vectors as features for hypernymy detection. Specifically, we use Support Vector Machine (SVM) [6] for this purpose. Inspired by the experiments of Julie Weeds et al. [29], several combinations of vectors are also experimental and reported.

Experiment 1. Experiment on $Vds1$ dataset, the data includes 976 hypernymy pairs (positive labels), and 1,026 pairs which are not hypernymy (negative labels), these pairs are mixed then selected 70% for training and 30% for testing. To increase the independence between training and testing sets, we exclude from the training set any pair of terms that has one word appearing in the testing set. The results shown in Table 5 are the accuracy of methods when using different combinations of vectors. The experimental results in Table 5 show that the improved method performs better than Word2Vec and DWN methods in terms of accuracy. svmDIFF gives better results for the Word2Vec model, but the performance of DWN and improved method is higher than with svmCATs.

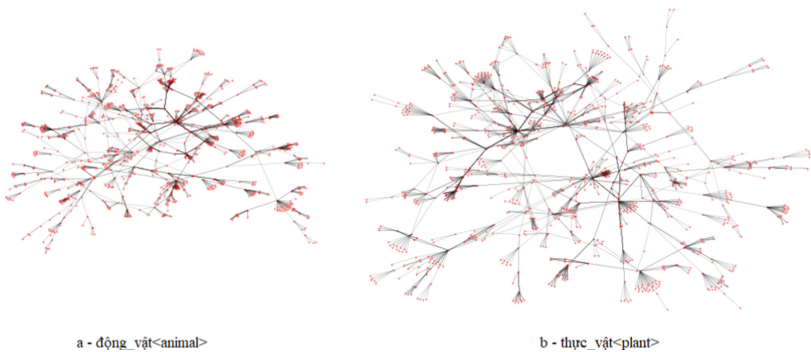


Fig. 3. Visualization of subnets.

Table 4. Statistics of three datasets.

Dataset	Relation	#Instance	Total	
Vds1	hypernymy	976	10285	
	co-hyponym	8283		
	Random	1026		
Vds2	hypernymy	1657	3314	
	Random	1657		
Vds3	động_vật<animal>	hypernymy	2284	2284
	thực_vật<plant>	hypernymy	2267	2267

Table 5. Hypernymy detection results for the Vds2 dataset.

Dataset	Model	svmDIFF	svmMULT	svmADD	svmCAT	svmCATs
Vds1	Word2Vec	0.82	0.77	0.81	0.80	0.79
	DWN	0.81	0.79	0.82	0.82	0.84
	Our	0.86	0.83	0.84	0.87	0.89

Experiment 2. Experiment on Vds2 dataset, the data includes 1,657 hypernymy pairs (positive labels), and 1,657 pairs which are not hypernymy (negative labels), the same as experiment 1, these pairs are mixed then selected 70% for training and 30% for testing. To increase the independence between training and testing sets, we exclude from the training set any pair of terms that has one word appearing in the testing set. The results shown in Table 6 are the performance of methods that are measured in terms of precision, recall, and F1.

Table 6. Hypernymy detection results for the Vds2 dataset.

Dataset	Model	Precision	Recall	F1
Vds2	Word2vec	0.85	0.87	0.86
	DWN	0.88	0.88	0.88
	Our	0.90	0.94	0.92

Experiment 3. This experiment aims to evaluate the capacity of methods to recognize a subnet. Two subnets: Vds3_{animal}, Vds3_{plant} respectively are used for training and testing data. In this experiment, svmCATs is used for combinations of vectors. Experimental results are presented in Table 7. In experiments 2 and 3, precision can be characterized as the measurement of exactness or quality, whereas recall is the measurement of completeness or quantity. As seen in Tables 6 and 7, the improved method produced better results than the original one, not only in terms of precision but also recall. Herein, the experiment focuses on Vietnamese hypernymy detection. However, our improved method can be easily adapted to other languages.

Table 7. Hypernymy detection results for the Vds3 dataset.

Model	Training	Testing	Precision	Recall	F1
Word2vec	Vds3animal	Vds3plant	0.50	0.60	0.55
DWN			0.52	0.64	0.57
Our			0.61	0.76	0.68
Word2vec	Vds3plant	Vds3animal	0.58	0.72	0.64
DWN			0.57	0.73	0.64
Our			0.62	0.78	0.69

The idea of incorporating semantic knowledge into the corpus-based learning of word embeddings has also been applied in the study of Quan Liu et al. [17]. Experimental results in [17] have shown that this approach can significantly improve the efficiency of NLP applications that rely on word embeddings.

5 Conclusion

A number of hypernymy detection methods based on word embeddings and supervised learning has been applied for Vietnamese. This paper offers two significant contributions. Firstly, a word embeddings model has been improved by weighting contextual words proportionally to the semantic similarity between them and the hypernym. Experimental results demonstrated that our proposal can increase the efficiency from 8% to 10% in terms of accuracy compared to the original method. Secondly, based on Vietnamese WordNet, three datasets for the Vietnamese hypernymy detection problem have been built and published. We intend to apply our method to detect other kinds of semantic relations and also other languages.

Acknowledgments. This paper is a part of project number KHCN-TB.23X/13-18 which is led by Assoc. Prof. Ngo Thanh Quy and funded by Vietnam National University, Hanoi under the Science and Technology Program for the Sustainable Development of Northwest Region.

References

1. Banerjee, S., Pedersen, T.: Extended gloss overlaps as a measure of semantic relatedness. In: Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, pp. 805–810 (2003)
2. Banerjee, S., Pedersen, T.: An adapted lesk algorithm for word sense disambiguation using wordnet. In: Gelbukh, A. (ed.) CICLing 2002. LNCS, vol. 2276, pp. 136–145. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45715-1_11
3. Baroni, M., Bernardi, R., Do, N.Q., Chieh Shan, C.: Entailment above the word level in distributional semantics. In: Daelemans, W., Lapata, M., Màrquez, L. (eds.) EACL, pp. 23–32. The Association for Computer Linguistics (2012)

4. Baroni, M., Lenci, A.: How we BLESSEd distributional semantic evaluation. In: Proceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics, pp. 1–10. Association for Computational Linguistics, Edinburgh, UK (2011). ‘aclanthology.org/W11-2501’
5. Biran, O., McKeown, K.R.: Classifying taxonomic relations between pairs of wikipedia articles. In: IJCNLP, pp. 788–794. Asian Federation of Natural Language Processing/ACL (2013)
6. Cortes, C., Vapnik, V.: Support-vector networks. *Mach. Learn.* **20**(3), 273–297 (1995)
7. Dagan, I., Roth, D., Sammons, M., Zanzotto, F.M.: Recognizing Textual Entailment: Models and Applications: Synthesis Lectures on Human Language Technology. Morgan & Claypool Publishers, San Rafael (2013)
8. Fellbaum, C. (ed.): WordNet: An Electronic Lexical Database. MIT Press, Cambridge (1998)
9. Fu, R., Guo, J., Qin, B., Che, W., Wang, H., Liu, T.: Learning semantic hierarchies via word embeddings. In: ACL, vol. 1, pp. 1199–1209. The Association for Computer Linguistics (2014)
10. Geffet, M., Dagan, I.: The distributional inclusion hypotheses and lexical entailment. In: Knight, K., Ng, H.T., Oflazer, K. (eds.) ACL, pp. 107–114. The Association for Computer Linguistics (2005)
11. Hearst, M.A.: Automatic acquisition of hyponyms from large text corpora. In: Proceedings of the 14th Conference on Computational Linguistics, COLING 1992, vol. 2, pp. 539–545. Association for Computational Linguistics, Stroudsburg (1992). <https://doi.org/10.3115/992133.992154>
12. Kennard, N.N.: Learning hypernymy over word embeddings (2015)
13. Kotlerman, L., Dagan, I., Szpektor, I., Zhitomirsky-Geffet, M.: Directional distributional similarity for lexical inference. *Nat. Lang. Eng.* **16**(4), 359–389 (2010)
14. Kozareva, Z., Hovy, E.: Learning arguments and supertypes of semantic relations using recursive patterns. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pp. 1482–1491. Association for Computational Linguistics, Uppsala (2010)
15. Lesk, M.: Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In: SIGDOC 1986: Proceedings of the 5th Annual International Conference on Systems Documentation, pp. 24–26. ACM, New York (1986). <http://doi.acm.org/10.1145/318723.318728>
16. Levy, O., Remus, S., Biemann, C., Dagan, I.: Do supervised distributional methods really learn lexical inference relations? In: Mihalcea, R., Chai, J.Y., Sarkar, A. (eds.) HLT-NAACL, pp. 970–976. The Association for Computational Linguistics (2015)
17. Liu, Q., Jiang, H., Wei, S., Ling, Z.H., Hu, Y.: Learning semantic word embeddings based on ordinal knowledge constraints. In: ACL, vol. 1, pp. 1501–1511. The Association for Computer Linguistics (2015)
18. Luu, A.T., Tay, Y., Hui, S.C., Ng, S.K.: Learning term embeddings for taxonomic relation identification using dynamic weighting neural network. In: Su, J., Carreras, X., Duh, K. (eds.) EMNLP, pp. 403–413. The Association for Computational Linguistics (2016)
19. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient Estimation of Word Representations in Vector Space. In: Proceedings of the First International Conference on Learning Representations, ICLR 2013, pp. 1–13 (2013)

20. Navigli, R., Ponzetto, S.P.: Babelnet: building a very large multilingual semantic network. In: Hajic, J., Carberry, S., Clark, S. (eds.) ACL, pp. 216–225. The Association for Computer Linguistics (2010)
21. Navigli, R., Velardi, P., Faralli, S.: A graph-based algorithm for inducing lexical taxonomies from scratch. In: Walsh, T. (ed.) IJCAI, pp. 1872–1877. IJCAI/AAAI (2011)
22. Nguyen, T.P., Pham, V.L., Nguyen, H.A., Vu, H.H., Tran, N.A., Truong, T.T.H.: A two-phase approach for building Vietnamese WordNet. In: Proceedings of the 8th Global WordNet Conference (GWC), pp. 261–266. Global Wordnet Association, Bucharest (2016). <https://aclanthology.org/2016.gwc-1.38/>
23. Roller, S., Erk, K., Boleda, G.: Inclusive yet selective: supervised distributional hypernymy detection. In: Hajic, J., Tsujii, J. (eds.) COLING, pp. 1025–1036. ACL (2014)
24. Seitner, J., et al.: A large database of hypernymy relations extracted from the web. In: Calzolari, N., et al. (eds.) LREC. European Language Resources Association (ELRA) (2016)
25. Snow, R., Jurafsky, D., Ng, A.Y.: Learning syntactic patterns for automatic hypernym discovery. In: NIPS, pp. 1297–1304 (2004)
26. Tan, B.V., Thai, N.P., Lam, P.V.: Construction of a word similarity dataset and evaluation of word similarity techniques for vietnamese. In: Nguyen, T.T., Le, A.P., Tojo, S., Nguyen, L.M., Phan, X.H. (eds.) KSE, pp. 65–70. IEEE (2017)
27. Tan, L., Gupta, R., van Genabith, J.: Usaar-wlv: hypernym generation with deep neural nets. In: Cer, D.M., Jurgens, D., Nakov, P., Zesch, T. (eds.) SemEval@NAACL-HLT, pp. 932–937. The Association for Computer Linguistics (2015)
28. Turney, P.D., Mohammad, S.M.: Experiments with three approaches to recognizing lexical entailment. *Nat. Lang. Eng.* **21**(3), 437–476 (2015)
29. Weeds, J., Clarke, D., Reffin, J., Weir, D.J., Keller, B.: Learning to distinguish hypernyms and co-hyponyms. In: Hajic, J., Tsujii, J. (eds.) COLING, pp. 2249–2259. ACL (2014)



Arbobanko - A Treebank for Esperanto

Eckhard Bick^(✉)

Institute of Language and Communication, University of Southern Denmark, Campusvej 55,
5230 Odense M, Denmark
eckhard.bick@mail.dk

Abstract. In this paper we describe and evaluate Arbobanko, a syntactic treebank for the artificial language Esperanto, as well as methods and tools used to produce the treebank. For an under-resourced language, the quality of automatic syntactic pre-annotation is of obvious importance, and by evaluating the parser associated with the treebank, we try to answer the question whether the language's extremely regular morphology and low lexical ambiguity carry over into a more regular syntax and higher parsing accuracy. On the linguistic side, the treebank allows us to address and quantify the typological issue of (free) word order in Esperanto.

Keywords: Treebanks · Esperanto · Dependency grammar · Constraint grammar · Syntactic parsing · Free word order languages

1 Introduction

Syntactic treebanks satisfy important needs in both language technology and descriptive linguistics, allowing the latter to identify and quantify linguistic patterns, and the former to train and evaluate machine-learned parsers. With a general change of focus from the latter to the former, dependency treebanks have become more prevalent at the expense of constituent treebanks, driven by methodological considerations such as implementability in mathematical models (graphs).

Historically, dependency syntax has roots in the description of slavic languages, one of its strengths being the handling of free word order and discontinuities, while constituent grammar was linked to English with its fixed word order and reliable subject-predicate pairs. Thus, the first and largest dependency treebank was built for Czech (Böhmová et al. 2003, Bejček et al. 2013), while major English treebanks like the Penn Treebank were originally annotated with phrase structure and converted to the dependency format only later (Johansson and Nugues 2007), by the machine-learning (ML) community. A third approach was used for the Danish Arboretum treebank (Bick 2003), where a rule-based Constraint Grammar (CG) parser was used to create shallow dependency trees that were then converted to constituent trees, using manual revision at both stages.

As an artificial language with a sizable living speaker community, Esperanto is a linguistically interesting language, albeit under-resourced in terms of both development/research funding and existing NLP resources. Our treebank project intends to address this issue at both the linguistic and NLP levels. We decided on a dependency format not only because of the current focus of the research community, but also because of the purported free word order-characteristics of the language, and because the only available parser was a CG dependency parser, and we needed to minimize (unfunded) human revision labor.

2 The Corpus

Arbobanko is a news corpus, covering the period 1997–2003. It is based on journalistic material from the Esperanto journal *Monato*, published by Flandra Esperanto Ligo, and compiled and TEI-encoded by Bertil Wennergren. The overall text corpus contains ca. 579,000 words, and is available for search and download at <http://tekstaro.com>. For the *Arbobanko* treebank a 50.000 word section of the corpus was tokenized and morphosyntactically annotated with the *EspGram* parser (Bick 2007 and 2009) and manually revised at all levels. Like the source corpus, this annotated subcorpus will be made available on-line.

Annotation was carried out with what could be called a recursive boot-strapping method, where corrections learned from manual revision were fed back into the parser in the form of rule changes or additions, that would then increase the accuracy of further automatic parses. By logging all manual corrections, it was also possible to establish an estimate of global and category-specific parser performance. Ultimately, knowledge of category-specific error margins should allow the use of a much larger treebank with only automatic annotation, that would still allow linguistic research with a reasonable level of reliability.

3 Annotation Levels

The treebank contains linguistic annotation at four primary levels: lemma, part-of-speech (POS) and inflexion, syntactic function (“edge labels”) and dependency-head id’s (attachment links). In addition, there is some secondary, lexical information about morpheme structure and POS-subclass, as well as some semantic class information, mostly for nouns. All information is strictly token-based and contained in the following ordered tag fields, with ‘@’ used as a marker for the syntactic label, and ‘#’ for a numbered dependency relation:

```
Wordformlemma < subclass >... POS inflexion @syntactic_function #head_id.
```

Apart from the linguistic annotation, most of the original TEI meta-information, such as topic, titles and paragraph id’s, is retained in the treebank on separate xml lines. In the example below, token lines were indented according to tree depth to increase readability. Apart from the native format, we also provide Tiger xml and the CoNLL tab field format with feature-attribute pairs.

Post	[post] <*> PRP @ADVL> #1->14	<i>After</i>
12	[12] <card> <cif> NUM P @>N #2->3	<i>12</i>
jaroj	[jaro] <dur> <per> N P NOM @P< #3->1	<i>years</i>
da	[da] PRP @N< #4->3	<i>of</i>
reformoj	[reformoj] <PREF:re%form o> <sem-c> <act> N P NOM @P< #5->4 <i>reform</i>	
la	[la] ART @>N #6->7	<i>the</i>
efikeco	[efikeco] <N:efik%ec o> <f> N S NOM @SUBJ> #7->14	<i>efficiency</i>
de	[de] PRP @N< #8->7	<i>efficiency</i>
la	[la] ART @>N #9->11	<i>the</i>
ĉehxa	[ĉehxa] <jnat> <Du> ADJ S NOM @>N #10->11	<i>czech</i>
ekonomio	[ekonomio] <domain> N S NOM @P< #11->8	<i>economy</i>
ne	[ne] <amod> <setop> ADV @>A #12->13	<i>not</i>
signife	[signife] ADV @ADVL> #13->14	<i>significantly</i>
transpaŝas	[transpaŝi] <PRP:trans+paŝ i> <mv> V PR VFIN @FS-STA #14->0 <i>surpasses</i>	
la	[la] ART @>N #15->16	<i>the</i>
nivelon	[nivelo] <ac> N S ACC @<ACC #16->14	<i>level</i>
atingitan	[atingi] <mv> V PCP PAS IMPF ADJ S ACC @ICL-N< #17->16 <i>achieved</i>	
en	[en] PRP @<ADVL #18->17	<i>in</i>
la	[la] ART @>N #19->20	<i>the</i>
jaro	[jaro] <dur> <per> N S NOM @P< #20->18	<i>year</i>
1989	[1989] <year> <card> <cif> NUM S @N< #21->20	<i>1989</i>
\$.	[.] PU @PU #22->14	.

[N=noun, ADJ=adjective, ADV=adverb, NUM=numeral, ART=article, PRP=preposition, V=verb, S=singular, P=plural, NOM=nominative, ACC=accusative, VFIN=finite verb, PR=present, IMPF=past, PCP=participle, @SUBJ=subject, @ACC=direct object, @ADVL=adverbial, @FS-STA=statement, @>N=prenominal, @N<=postnominal, @>A=pre-adject, @P<=argument of preposition, @ICL-N<=postnominal non-finite clause]

3.1 Morphological Annotation

A low degree of morphological ambiguity is a planned design feature of Esperanto, and together with its regular inflexion and affixation system, meant to make the language easy to learn. As a result, automatic annotation is very reliable at this level, and few ambiguity classes exist, with little need for human revision. The only systematic POS ambiguity is between proper nouns and other word classes because of upper-casing (especially in sentence-initial position), and in connection with tokenization errors. Thus, the otherwise reliable vowel coding for POS (e.g. -o = noun, -a = adjective, -i = infinitive, -e = adverb) breaks down in the face of foreign names in (a) and (b). Another type of ambiguity arises from the syntactic, rather than morphological, nature of some non-inflecting word classes (c1–3).

- (a) Durrës-Varna -- > adjective -a
- (b) Verdi kaj Ĉajkovskij -- > verb -i
- (c1) ĝis la mateno [until morning], -- > preposition
- (c2) ĝis ili subskribis [until they signed], -- > conjunction
- (c3) ĝis kvar gastoj [up to four guests], -- > adverb
- (d) DNA, RNA -- > proper?/noun
- (e) i.a. [among other things] -- > noun?/adverb

Sometimes, abbreviations can also present problems, because of upper-casing and lack of endings: type (d) is sometimes mis-tagged as e.g. company proper nouns, and dot-shortened abbreviations may default to a (wrong) noun reading.

A final, rare type of ambiguity concerns morpheme structure, and is a source of puns in Esperanto. Although this ambiguity class will not be visible at the lemma/POS/inflexion level, it does affect the meaning of a word, and the *EspGram* parser tries to resolve it (f-g).

(f) *altiri* < *ADJ:alt + *irli* > (“go high” [high + go]) vs. < PRP:al + *tirli* > (“attract” [to-draw])

(g) *diamante* < *N:di + *amante* > (“God-lovingly”) vs. < *ADJ:dila + *mante* > [“godly-mantis-ly”] vs. un-compounded “diamond-like”

In principle, there is no inflectional ambiguity in Esperanto. However, foreign proper nouns that have not been assimilated into the language, often retain their original spelling and will rarely receive the accusative case marker -n, unless they happen to end in -o (the noun-marking vowel). Therefore, such proper nouns are nominative/accusative-ambiguous and a theoretical source of errors for *EspGram*’s disambiguation.

3.2 Syntactic Annotation

Syntactic annotation is of course, what a treebank is really about. Thus, the linguistic motivation for creating *Arbobanko* is to allow descriptive studies of Esperanto syntax, addressing topics such as word order and structural complexity. It is for such linguistic reasons, that the relatively fine-grained syntactic tag inventory of *EspGram* is maintained in the treebank. For instance, what could have been one adverbial class, is subdivided into free adverbials (@ADVL), bound adverbials (@SA), object-bound adverbials (@OA), free predicatives (@PRED) and prepositional objects (@PIV). In noun phrases, a distinction is made between *identifying* (@APP) and *predicating* (@N < PRED) appositions. However, we try to avoid unnecessary tag complexity by not introducing different syntactic tags, where POS already contains the distinction. Thus, phrase-level modifiers are only attachment-tagged as prenominals (@ > N) and postnominals (@N <) nominals, or pre-adjectives¹ (@ > A) and post-adjectives (@A <), not for what the modifier itself is (e.g. hypothetical @nmod for a modifier that is a nouns), because that would just be duplicated information.

In the same vein, a strict form-function distinction is maintained for dependency heads. For instance, adjectives are not re-tagged as nouns, just because they appear as the head of a noun phrase. In English translation, “sick” stays ADJ in “the sick flocked to him”, in spite of it being the head of the subject np. This way, there will be no conflict in it taking an adverb modifier (“the very sick flocked to him”), because “very” still can see necessary ADJ head to attach to. The “noun-ness” of “sick” in “the sick” will thus be expressed solely at the function level, by it carrying a noun function (subject) and an article dependent.

¹ Adjectives are defined as adverbial modifiers in *adjp*’s and *advp*’s, i.e. of adjectives and adverbs.

While the above adjective-noun duality is often avoided in Esperanto by adding POS-changing suffixes (mal-san-**ul**-o = un-healthy-person-noun), another word class, participles, is more problematic, having both adjectival and verbal aspects. Esperanto adjectival participles inflect in gender and number, but are also marked for tense/aspect [aio] and passive/active [\pm n], and often function as non-finite predicators with one or several verb arguments. Therefore, even though there is only one morphological (“form”) analysis, the ambiguity manifests at the syntactic function level and needs to be resolved contextually (a-b).

(a) *numeritaj* biletoj [numbered tickets] -- > @ > N (prenominal)

(b) transportkoridoroj *numeritaj* per romaj ciferoj [traffic corridors numbered with Roman numerals] -- > @ICL-N < (postnominal [N <] non-finite [I] clause [CL])

3.3 Dependency Annotation

In a typical Constraint Grammar parsing chain, each linguistic level will receive its own grammar module, and disambiguated output from one will be used as input to the next. Classical CG (Karlsson 1990) recognizes three levels: Morphological/POS disambiguation, syntactic function mapping (e.g. based on case or position), and syntactic disambiguation. Syntactic form (structural tags) was addressed only rudimentarily, with arrows indicating attachment direction (e.g. @N < pointing left to a noun head). The state-of-the-art CG3 compiler (Bick and Didriksen 2014) does expand the formalism to allow the creation and use of dependency links, but with pre-existing morphosyntactic parsers this will mean a dependency module that is run *after* function labels have already been assigned - a design different from most machine-learning (ML) approaches, such as the ones described in the CoNLL conference joint tasks on dependency parsing (e.g. Nivre et al. 2007), that will perform the two tasks simultaneously or in the opposite order. This function-first architecture of our automatic annotation system means that dependency attachment rules can exploit existing syntactic information (including attachment direction!), but it also means that many attachment errors need to be fixed in *EspGram* itself, rather than in the add-on dependency module.

In descriptive terms, our native dependency annotation is syntactically motivated rather than semantic, minimizing the dependency distance between a governing head and the token it controls in terms of agreement or valency. Thus, prepositions are treated as heads of pp’s, because the verbs and nouns governing the pp may have preposition-specific valency (e.g. *rilati al* [refer to], *amikeco kun* [friendship with]). In the same vein, auxiliaries are regarded as (syntactic) heads of verb chains, because they control the form of the main verb (infinitive, participle), rather than vice versa. We are aware of the Universal Dependencies (UD) initiative (McDonald, et al. 2013) that uses semantic head relations instead (i.e. prepositions and auxiliaries as dependents of main verbs and pp-nouns, respectively), but have chosen to keep syntactic and semantic levels strictly separate in *Arbobanko*. Semantic argument links with thus be added only in a future version with full semantic role and frame annotation. That said, we provide an automatically UD-converted version of the treebank in CoNLL format to further comparability and to allow compatibility with UD-based NLP tools.

Two notoriously difficult issues for a dependency grammar are coordination and ellipsis, both because dependency grammar does not allow empty nodes, forcing either (a) parallel attachment with a loss of structural information or (b) some kind of "dependent nexus", where one dependent attaches to another rather than the common antecedent. (a) provides short semantic paths for all constituents, but we opted for (b) in the default version of the treebank, again giving priority to syntactic concerns and expliciting the special relation between conjuncts and the parts of an elliptic nexus, respectively. However, sequential attachments of second and later conjuncts to the first conjunct can easily, and automatically, be raised to parallel attachment, if corpus users wish to use the latter format.

Finally, we have chosen to include punctuation in our dependency mark-up. Paired punctuation (e.g. parentheses) will attach to the highest node in the enclosure, and clause and phrase separators attach left to the highest node of the preceding clause or phrase.

3.4 Secondary Tags

Secondary tags are marked with `<... >` brackets and comprise secondary grammatical and semantic information. They can be mapped either from a lexicon file or by contextual CG rules. Grammatical tags will be disambiguated contextually, if more than one of the same type is possible for a given word, for instance the distinction between `< rel >` (relative) and `< interr >` (interrogative) for adverbs and pronouns, or `< mv >` (main verb) and `< aux >` (auxiliary) for verbs. A third type of secondary tag helps with coordination conversion: `< cjt-first >` (first conjunct), `< cjt >` (second or later conjuncts) and `< co-arg >` for coordinating conjunctions, with "arg" specifying the syntactic tag of the coordinated material, e.g. `< co-subj >` for subject coordination.

Semantic tags at this level are not functional (semantic roles), but lexical (ontology-derived), and they are not currently disambiguated in the treebank. Most tags belong to a shallow noun ontology of about 200 classes² and help *EspGram*'s CG rules to e.g. choose subject readings over other possible syntactic tags, if a word belongs to a human class (e.g. `< Hprof >` professional, `< Hnat >` nationality term, `< Hideo >` ideology-follower).

4 Parser Evaluation

The focus of this paper is on the creation of a treebank for a language, where there was none, i.e. the resource side rather than the performance side of NLP. So we have evaluated the underlying parser not for its own sake, but in order to be able to improve it and thereby speed up further manual revision of the treebank. Also, category-specific accuracy is useful when interpreting linguistic results from larger, unrevised treebanks made with the same parser.

² For Esperanto, we have adopted the "semantic prototype" ontology described at http://visl.sdu.dk/semantic_prototypes_overview.pdf.

Our evaluation is based on the change log from the manual revision of the first 16300 tokens of the treebank. Because attachment errors were counted separately, attachment direction arrows at the clause level were ignored when evaluating function tags (i.e. @ < SUBJ and @SUBJ > were both counted as just @SUBJ, subject). This evaluation method is clearly more lenient than an independent gold corpus or an independent manual annotation of the same corpus would have been, because when in doubt, a reviewer-annotator will simply choose to do nothing and leave the automatic tag unchanged. A positive side effect of this parser bias, however, is a certain consistency with regard to the resolution of dubious cases, derived from the reproducibility of the automatic choice, and difficult to achieve for human annotators. Also, the parser bias will only affect unclear cases, and still produce good statistics for safe errors, allowing us to flag the most error-prone categories for further inspection.

All in all, ca. 3% of tokens in the test section had errors in primary categories, with 2.6% attachment errors and 1.6% function tag errors. Performance for word tokens alone (ignoring punctuation) is shown in parentheses in Table 1. As expected for Esperanto, the extremely regular morphology left almost no room for POS or inflexion errors.

Table 1. Parser performance

	Correct attachment	Wrong attachment	
Correct function	97.04% (96.53)	1.36% (1.56)	98.40% (98.09)
Wrong function	0.35% (0.42)	1.25% (1.49)	1.60% (1.91)
	97.39% (96.95)	2.61% (3.55)	100% (100)

In a breakdown of individual categories (Table 2) pp-attachment problems left their predictable mark, with postnominal pp's (PRP @N <) being attached to the wrong noun, or tagged as adverbial (@ADVL) and attached to a verb. Thus, 19.8% of attachment errors and 26.8% of function errors involved the postnominal category, and 90% of cases were pp's. If predicating appositions are included in this category, it comprises 1/4 - 1/3 of all errors.

Table 2 contains only the major categories, and it lumps all clause functions into only two groups, finite and non-finite, but it clearly shows what is difficult for function tagging and for attachment tagging, respectively. Thus, coordinators (@CO) and, to a lesser degree, adverbials (@ADVL) are more an attachment than a labeling problem, while copula complements (@SC) and subjects (@SUBJ) are more a labeling than an attachment problem. For postnominals (@N <, @N < PRED), a function error will almost always lead to an attachment error, but the latter bears the additional burden of distance errors. That direct objects (@ACC) are so easy to label, is due to the fact that Esperanto has a morphological accusative marker (-n).

Table 2. Error contribution by category (accuracy)

	% of function errors	% of attachment errors	% of all tokens
@N < (postnominal)	26.8	19.8	5.6
@N < PRED (predicat. Apposition)	7.3	6.8	0.9
@ADVL > (left adverbial)	5.4	6.8	5.9
@ < ADVL (right adverbial)	3.8	5.2	4.8
@SUBJ (subject)	6.9	4.0	8.8
@ACC (direct object)	1.9	3.1	4.8
@SC/@SA (copula complements)	3.8	0.7	1.9
@NPHR (free np, no verb)	5.0	2.1	0.6
@A < (post-adject)	3.4	2.1	5.2
@FS-... (finite clauses)	15.0	16.7	9.5
@FS-N < (relative clause)	2.7	6.1	1.3
@ICL-... (non-finite clauses)	6.1	5.2	2.9
@CO (coordinator)	1.1	12.9	3.3
@PU (punctuation)	0.0	0.1	16.2

While Table 2 tells us, where errors occur most in absolute terms, and where added revision and rule-writing should be focused for maximal treebanking efficiency, this is not enough to predict which linguistic information weaned from the corpus is reliable and which is not. For this task, error rates need to be normalized with regard to overall category frequencies. Figure 1 models this category-specific error risk computed as error share divided by token share. The resulting value tells us, how much a category is overrepresented among errors as compared to its share among running tokens. Thus, the most unreliable categories in terms of function labeling are @N < PRED and @NPHR (9 times overrepresented), while all clause-level categories with the exception of complements, i.e. subjects, objects and adverbials are safe (underrepresented).

In terms of attachment, the most unreliable categories are coordinators (@CO) and relative clauses (@FS-N <), with a 4x over-representation, and the same np categories that are also unreliable in terms of function (@N <, @N < PRED and @NPHR). All in all, Fig. 1 predicts that an automatically annotated treebank is safer to use for clause-level studies than np-level studies and coordination studies.

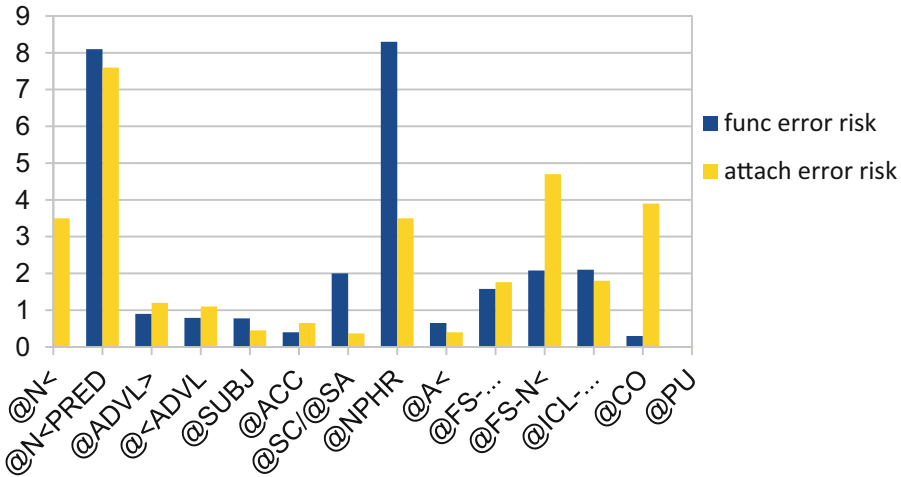


Fig. 1. Error risk by category

5 Linguistic Evaluation

Our first research question was methodological: Does the regular morphology of Esperanto spill over into a more regular syntax in the sense, that parsing will be easier? With our data, the answer to this question appears to be only a little yes. The morphological error rate was indeed very low, but syntactic accuracy (~96.5% for word tokens) is only marginally better than what has been reported for CG systems for other languages (e.g. 95–96% for Portuguese [Bick 2014]). Also, recall results for English CG (Prytz 1998) indicate that printed news are probably situated at the high performance end, and that other genres would likely fare worse. Especially the problems with pp attachment and coordination indicate that at the syntactic level, Esperanto is not so different from other languages, and that ambiguity in this area arises from semantics rather than morphology.

The second research question is linguistic - to what degree does Esperanto have free word order? At the clause level, data from Arbobanko indicate a general tendency towards SVO order, but also category-specific deviations. For the statistics in Table 3, relative and interrogative pronouns were excluded because they are always used clause-initially, irrespectively of syntactic category³, in both Esperanto and all etymologically related languages.

As can be seen, subjects and direct objects occur on the “wrong” side of the verb often enough to speak of free word order in the sense that such usage is grammatically acceptable, though not the default, in Esperanto. For oblique and copula arguments, however, left placements (outside relative clauses and questions) is so rare, that it should be considered as marked (e.g. focus-triggered). Object *pronouns* were as (un)likely as

³ Yes/no questions with the question particle “ĉu” were not excluded, but were not statistically salient, because only a few contained finite verbs.

Table 3. Clause level constituent placement

	Left of V	Right of V
Subject (@SUBJ)	85.4%	14.6%
Direct object (@ACC)	10.2%	89.8%
Copula complement (@SC)	3.6%	96.4%
pp/oblique object (@PIV)	2.7%	97.3%

object *nouns* to occur left of the verb, so clitic effects in the fashion of Romance languages can be ruled out.

In order to identify complete finite clauses with both subject and object, and count full SVO patterns, we wrote a small mark-up CG mapping %svo, %vso, %sov etc. tags on the main verbs of these clauses (Table 4).

Table 4. SVO variations

	Percentage of finite clauses with both subject and direct object
SVO	89.98%
OVS	2.44%
SOV	2.69%
OSV	3.42%
VSO	-
VOS	1.47%

The numbers indicate that SVO *is* the default word order for Esperanto outside relative clauses and questions, but that there is no strict rule against other word orders, that together make up 10% of finite S + O clauses. Only VSO did not occur at all. Unexpectedly, the “Yoda” word order OSV is the *most* frequent alternative, in spite of it being the only one that is not documented as a normal word order in natural languages. Non-finite clauses⁴ had a stricter word order than finite clauses, with 98% of objects placed to the right.

At the phrase level, the typologically interesting word-order question is where adjectives are placed in noun phrases. Here, our data did contain some variation, with left placement as the statistical norm, but still 5.9% of adjectives positioned right of their head noun. In addition, heavy modifier material seems to be moved to the right. Thus all modifier clauses, including participle clauses, were placed to the right, as well as half of the coordinated adjectival modifiers.

⁴ Non-finite clauses do not take subjects in Esperanto.

One conclusion from our np word order data is that Esperanto, despite the fact that the majority of its vocabulary can be traced back to Romance languages, seems to prefer a “Germanic”, left placement of adjectives. We therefore also investigated verb phrases, looking for discontinuities, common in Germanic languages. But while we did find about 15.3% discontinuous vp’s, almost all interfering material was adverbial, with no sign of post-auxiliary subjects, occurring in many Germanic languages when fronting other constituents.

The last linguistic topic we will present here is the use of complex tense, mode and aspect. For this, Esperanto combines the tense-inflected *esti* (“be”) with likewise tense-inflected active and passive participles⁵. Because of the rareness of some combinations, and the low error rate of automatic annotation for this type of auxiliary construction, we have used the entire Monato corpus, with automatic treebank annotation, for the data in Table 5.

Table 5. Auxiliary constructions

AUXILIARY:	estas (present)	estis (past)	estos (future)	estus (conditional)
ACTIVE PCP:				
-anta (present, "...-ing")	5	3	-	-
-inta (past, "having ...-ed")	7	24	1	13
-onta (future, "going to ...")	1	2	-	1
PASSIVE PCP:				
-ata (present, imperfective, "being ...-ed")	176	79	17	3
-ita (past, perfective, "having been ...-ed")	231	244	30	9
-ota (future, prospective, "to be ...-ed")	2	1	-	-

As can be seen, passives are much more common than actives, probably because the latter cover less linguistic terrain and “only” work as complex tenses, with a “viewer” time marked by the auxiliary, and a relative event time marked as anterior, posterior or simultaneous in the participle. The high-frequency complex passives (*estas/estis/estos* + ...*ata/ita*), on the other hand, are the only way to express finite passives, since only actives have auxiliary-free finite forms. In addition, the participle tense vowel in these forms is used to express aspect (*a*/present = imperfective, *i*/past = perfective). The conditional auxiliary form *estus* (last column) is rarest, an mostly used for past conditionals, active or passive. The active present participle is rare, and never used with future and conditional *estos/estus*, implying that no added meaning is achieved compared to the *-os/us* forms of the main on its own.

⁵ These participles carry an adjectival *-a* ending, and inflect/agree with regard to number and case, allowing them to function as postnominal non-finite clauses, marked @ICL-N < in the treebank, unlike the @ICL-AUX < (argument of auxiliary) we are concerned with here.

6 Conclusion and Outlook

We have presented and evaluated a treebank for Esperanto, that we hope will help remedy the lack of NLP resources for the language and trigger further research. By constantly improving the grammar and lexicon of the underlying CG parser, manual revision labour was kept at a minimum. Measured against the revised annotation, the parser achieved a syntactic accuracy (labelling and attachment combined) of 96.5% for non-punctuation tokens, albeit with considerable variation across categories. This relatively high performance should facilitate future work that could include a “raw” (automatic) treebank for the entire *Monato* corpus, as well as new treebank sections for other genres.

On the linguistic side, the treebank has allowed us to establish word (constituent) order statistics classifying Esperanto as an SVO and ADJ-N language with considerable room for word order variation, both at the clause level and for np attributes. What we do not know, and what should be addressed in future research, is to which degree these findings depend on statistical tendencies influenced by the native language of an Esperanto speaker/author, and whether word order variation is less or more pronounced in the formal written language of a news journal than in spoken or informal written language, as found in social media, e-mail or text messages.





References

1. Bejček, E., Hajičová, E., Hajič, J., et al.: Prague Dependency Treebank 3.0. (Data/Software). Charles University in Prague, MFF, ÚFAL. [<http://ufal.mff.cuni.cz/pdt3.0/>] (2013)
2. Bick, E., Didriksen, T.: CG-3 - beyond classical constraint grammar. In: Beáta, M., Proceedings of NODALIDA 2015, May 11–13, 2015, Vilnius, Lithuania, pp. 31–39. LiU Electronic Press, Linköping (2015)
3. Bick, E.: PALAVRAS, a Constraint Grammar-based Parsing System for Portuguese. In: Tony Berber, S., de Lurdes São Bento Ferreira, T. (eds.), Working with Portuguese Corpora, pp. 279–302. Bloomsbury Academic, London/New York (2014)
4. Bick, E.: A Dependency Constraint Grammar for Esperanto. Constraint Grammar Workshop at NODALIDA 2009, Odense. NEALT Proceedings Series, Vol. 8, pp. 8–12. Tartu: Tartu University Library (2009)
5. Bick, E.: Tagging and parsing an artificial language: an annotated web-corpus of Esperanto. In: Proceedings of Corpus Linguistics 2007, Birmingham, UK. [<http://ucrel.lancs.ac.uk/publications/CL2007/>] (2007)
6. Bick, E.: Arboretum, a Hybrid Treebank for Danish. In: Joakim, N., Hinrich, E. (eds.) Proceedings of TLT 2003 (2nd Workshop on Treebanks and Linguistic Theory, Växjö, 14–15 November 2003), pp. 9–20. Växjö University Press (2003)
7. Böhmová, A., Hajič, J., Panenová, B.H.J., Hajicova, E.: The Prague dependency treebank: a 3-level annotation scenario. In: Abeillé, A. (ed.) Treebanks: Building and Using Parsed Corpora. Dordrecht, the Netherlands: Kluwer, pp. 103–126 (2003)
8. Johansson, R., Nugues, P.: Extended constituent-to-dependency conversion for English. In: Proceedings of NODALIDA 2007. Tartu, Estonia (2007)
9. McDonald, R., et al.: Universal dependency annotation for multilingual parsing. In: Proceedings of ACL 2013 (2013)

10. Karlsson, F.: Constraint grammar as a framework for parsing running text. In: Proceedings of the 13th Conference on Computational, vol. 3, pp. 168–173. ACL (1990)
11. Nivre, J., et al.: The CoNLL 2007 shared task on dependency parsing. In: Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007, pp. 915–932 (2007)
12. Prytz, K.: Evaluation of the syntactic parsing performed by the ENGCG parser. In: Proceedings of the 11th Nordic Conference on Computational Linguistics, Copenhagen, 28–29 January 1998. ACL web anthology (1998)



Sentence Compression as a Supervised Learning with a Rich Feature Space

Elena Churkin¹, Mark Last² , Marina Litvak¹ , and Natalia Vanetik¹  

¹ Department of Software Engineering, Shamon Academic College, Beer Sheva, Israel
{marinal,natalyav}@sce.ac.il

² Department of Software and Information Systems Engineering, Ben Gurion University of the Negev, Beer Sheva, Israel
mlast@bgu.ac.il

Abstract. We present a novel supervised approach to sentence compression, based on classification and removal of word sequences generated from subtrees of the original sentence dependency tree. Our system may use any known classifier like Support Vector Machines or Logistic Model Tree to identify word sequences that can be removed without compromising the grammatical correctness of the compressed sentence. We trained our system using several classifiers on a small annotated dataset of 100 sentences, which included around 1500 manually labeled subtrees (removal candidates) represented by 25 features. The highest cross-validation classification accuracy of 80% was obtained with the SMO (Normalized Poly Kernel) algorithm. We evaluated the readability and the informativeness of the sentences compressed by the SMO-based classification model with the help of human raters using a separate benchmark dataset of 200 sentences.

Keywords: Sentence compression · Syntactic dependencies · Supervised learning

1 Introduction

Sentence compression is a common NLP task of shortening sentences without changing their meaning and while preserving their correct grammatical structure. One of the important applications of sentence compression is automatic text summarization [11, 14]. The other known applications are headline generation [5], generation of television subtitles [18], automatic tweet generation [17], and displaying texts on small screens like mobile phones [6]. Most approaches to the sentence compression task are based on removing words from the sentences, but systems that use paraphrasing also exist [4]. Common sentence compression methods include integer linear programming [2], noisy-channel models [10, 13], models that use pruning of dependency and constituency parse trees [1, 12] and discriminative large margin learning [3, 16]. One of the latest sentence compression methods is based on a probabilistic model (LSTM - Long Short Term Memory) and it does not utilize any syntactic information (like PoS or parse trees) nor the desired compression length to compress sentences [7]. However,

it builds upon a large set of features and thus requires a training corpus of considerable size. In [7], the LSTM system is trained on two million sentence-compression instances.

In this paper, we present a new supervised sentence compression method based on detection, labeling, and removing appropriate word sequences (subtrees) from the original sentence. Similar to the cited works, our approach uses the dependency structure of a sentence and prunes word sequences corresponding to dependency subtrees. However, our approach is supervised and does not contain any ad-hoc rules. The subtrees of the dependency parse tree are treated as removal candidates. For each subtree, we calculate 25 predictive features that are later filtered by a feature selection method. Our system may use one of the known classification methods (like SVM) to identify subtrees that can be removed without compromising the grammatical correctness of the compressed sentence. Due to a limited amount of predictive features, our system may be trained on a much smaller set of compressed sentences than LSTM-based algorithms. Moreover, the system compression ratio may be controlled by the user. We trained our system with several classifiers on a manually labeled dataset of 1494 subtrees extracted from 100 sentences and evaluated it on a separate test set of 200 sentences. Using dependency tree pruning for sentence compression is motivated by the belief that the grammatical correctness of the compressed sentences can be better ensured by pruning of dependency trees because tree pruning approaches do not generate new dependencies and are unlikely to produce a compression with a different meaning [8]. An unsupervised dependency tree pruning approach is used in [8], whereas in this paper we propose a new supervised tree pruning approach for sentence compression. The current version of our system has been trained on English texts but in principle, it can be trained on an annotated collection of parsed sentences in any language.

2 The Compression Methodology

Our sentence compression methodology is composed of several stages: (1) extraction of removal candidates (word sequences represented by subtrees of a dependency parsing tree), (2) representation of extracted candidates by predictive features (including feature extraction and selection), and (3) candidates classification (valid/invalid for removal) using a pre-trained classification model. Figure 1 demonstrates the general pipeline of our compression procedure.

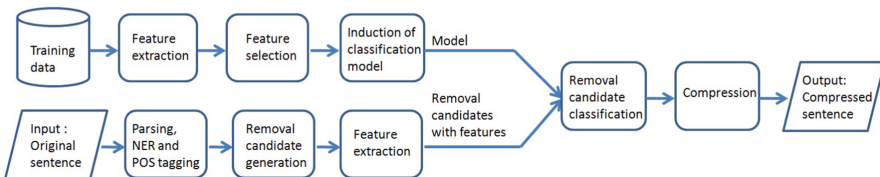


Fig. 1. Pipeline of our sentence compression tool.

2.1 Generation of Subtrees (Removal Candidates)

For each sentence, our system identifies as removal candidates all word sequences representing the possible subtrees of the dependency syntax tree.¹ Also, subtrees' complements and single words (remaining after filtering) are added to the list of candidates. The dependency syntax trees are generated using the Stanford Parser from the Stanford coreNLP tool [15].

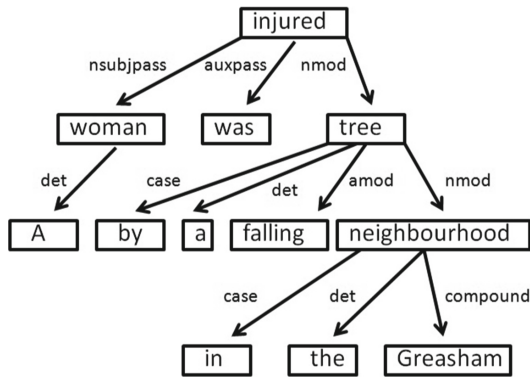


Fig. 2. Dependency syntax tree representing the sentence “A woman was injured by a falling tree in the Gresham neighborhood”.

An example of a dependency syntax tree for the sentence “A woman was injured by a falling tree in the Gresham neighborhood” is demonstrated in Fig. 2 and selected removal candidates are shown in Table 1, along with the compressed sentences after removal of the corresponding subtree, its type (single word, subtree, or its complement) and its class label (“Y” - valid, “N” - invalid). The sentence has 11 subtrees and 12 single words. Determiners, prepositions, and “be” verbs are filtered, retaining the set of only five removal candidates and their complements for annotation or classification.

Table 1. Some possible candidates with their class labels (manually annotated).

Removal candidate	Compressed sentence	Type	Class
Falling	A woman was injured by a tree in the Gresham neighborhood	word	Y
By a falling tree in the Gresham neighborhood	A woman was injured	Subtree	Y
A woman was injured	By a falling tree in the Gresham neighborhood	Complement	N

¹ Complete sentences and single words that are prepositions, wh-words, pronouns, and forms of the “to be” verb are not saved in the list of removal candidates.

2.2 Feature Extraction

For each removal candidate, we calculate 25 different features which are supposed to describe its complete grammatical structure. Some of the features are obtained using the Stanford coreNLP tool. We explored four different categories of features: based on statistical information, dependency and constituency parsing, Named Entity Recognition (NER), and Part-Of-Speech (POS) tagging. The full list of features is as follows.

1. General quantitative features

length: Number of tokens in the subtree.

ratio: Number of tokens in the subtree, normalized by the sentence length.

rel_start, *rel_end*: Relative location of the first/last subtree word in a sentence.

chars: Total number of characters in the subtree's words.

2. Syntax features²

dependency: Grammatical relation of the subtree's parent to the subtree root in the dependency parsing tree. In order to avoid overfitting, we group all dependency relations to nine groups: "core", "noncore", "spec", "noun", "comp", "coord", "case", "loose", and "other".

depth: Distance between the subtree and the root of the dependency syntax tree.

parent: Part of speech (POS) of the parent of the subtree in the dependency tree.

subj: Does a subtree contain a subject? (true/false)

verb: Does a subtree contain a verb? (true/false)

obj: Does a subtree contain an object? (true/false)

phrase_type: Phrase type of the subtree according to the sentence constituency syntax tree.³ *punct_in*: Does a subtree contain punctuation marks? (true/false)

punct_out: Is a subtree surrounded by punctuation marks? (true/false)

3. NER features

is_ner: is a subtree a named entity (NE)?

ner_type: NE type of a subtree.⁴

4. POS features

POS_before, *POS_after*: POS label of the last word preceding and the first word following the word sequence represented by the subtree.

POS_first, *POS_last*: POS label of the first/last word in the word sequence.

%nouns, *%verbs*, *%adjectives*, *%adverbs*, *%prepos*: Percentage of nouns, verbs, adjectives, adverbs, or prepositions in the subtree, respectively.

Feature selection is performed before applying a classification algorithm to the labeled data (see the next section).

2.3 Compression Algorithm

Algorithm 1 contains the pseudocode of the proposed method. The inputs of the algorithm are a set of complete sentences S to be compressed, a classification algorithm

² All syntax features are calculated using the dependency syntax tree, except for the last one, is obtained from the constituency parse tree.

³ We use "LEAF" type for a single word.

⁴ We use "null" for the non-NEs.

Algo, and a manually annotated dataset D (with already selected features). The output of the algorithm is a set of compressed sentences C . First, a classification model M is induced from the dataset D using the algorithm *Algo*. For each sentence, S_i from the set S , a dependency parse tree is created (using Stanford parser) and all subtrees of the dependency tree are identified, filtered (determiners, preposition, “wh”-words, etc.), and saved in the set of removal candidates ST_i . Every subtree (removal candidate) is converted to a sequence of words, the features (depending on the classification algorithm *Algo*) are extracted, and the model M is used to classify this candidate. All candidates that are classified as “Y” by M are removed from the corresponding sentence and the compressed sentences are added to the output set C .

For example, if we run the algorithm on single sentence “A woman was injured by a falling tree in the Gresham neighborhood”, it detects (classify to “Y”) the following removing candidates: “by a falling tree in the Gresham neighborhood”, “in the Gresham neighborhood”, “falling”.

Algorithm 1: Sentence compression

```

Input: set  $S$  of original sentences  $S_1, \dots, S_n$ 
          classification algorithm Algo
          annotated dataset  $D$ 

Output: set  $C$  of compressed sentences
 $M \leftarrow \text{induce\_model}(D, \text{Algo})$ 
 $C \leftarrow \emptyset$ 
foreach  $S_i \in S$  do
  |  $DT_i \leftarrow \text{generate\_dep\_tree}(S_i)$ 
  |  $ST_i \leftarrow \text{all\_subtrees}(DT_i)$ 
  |  $ST_i \leftarrow \text{filter}(ST_i)$ 
  |  $Cands_i \leftarrow \emptyset$ 
  | foreach  $ST_{ij} \in ST_i$  do
  | |  $Cand_{ij} \leftarrow \text{get\_words}(ST_{ij})$ 
  | |  $Attr_{ij} \leftarrow \text{attributes}(Cand_{ij})$ 
  | |  $Class \leftarrow \text{classify}(Attr_{ij}, M)$ 
  | | if  $Class = Y$  then
  | | |  $Cands_i \leftarrow Cands_i \cup Cand_{ij}$ 
  | | end
  | end
  |  $C_i \leftarrow S_i$ 
  | foreach  $Cand_{ij} \in Cands_i$  do
  | |  $C_i \leftarrow C_i - Cand_{ij}$ 
  | end
  |  $C \leftarrow C \cup C_i$ 
end
return  $C$ 

```

All stages of our algorithm, except classification, are polynomial (quadratic) in the dependency tree size (number of its nodes). Practically, the most computationally expensive part of our compression pipeline is classifying the removal candidates by a specific classification model (e.g., SMO-NPK).

3 Experiments

We performed both automated and human evaluations of the proposed method. Automated evaluations were aimed at selecting the most accurate subtree classifier for our compression procedure. The purpose of human evaluations was to estimate the quality of the compression and compare our approach to the state-of-the-art method [7].

3.1 Training Data

Our training data is composed of a random sample of 100 sentences from DUC 2002⁵, with various lengths, number of clauses, and amount of punctuation marks. Four annotators labeled the extracted removal candidates as “valid” if the corresponding word sequences could be removed without compromising the sentence meaning and its grammatical correctness. Finally, we labeled with “Y” the subtrees marked by the majority of annotators as “valid” and with “N” all other subtrees. The resulting training dataset contains 1494 subtrees (690 “Y” and 804 “N” instances) extracted from 100 sentences.

3.2 Classifiers

Using the Weka tool [9], we evaluated 14 different classification algorithms and finally chose four classifiers that performed best with our dataset, namely: Logistic Regression (LR), Sequential Minimal Optimization with Normalized Poly Kernel (SMO-NPK), Classification Via Regression (CVR), and Logistic Model Trees (LMT).

Table 2 shows the subtree classification performance, measured by the precision and the recall of the “Y” class using 10-fold cross-validation and all 25 features.

In order to simplify the trained models, shorten training times, and reduce overfitting, we performed feature selection with the following methods: iterative subset selection with Gain Ratio and Info Gain (bottom-up) and backward elimination with “Y” precision (top-down).

Table 2. Evaluation of various classifiers.

Classifier	Precision	Recall	F 0.5
LR	0.750	0.754	0.751
SMO-NPK	0.766	0.741	0.761
CVR	0.739	0.761	0.743
LMT	0.737	0.771	0.744

Performance of feature selection methods is shown in Figs. 3, 4 and 5.

The best “Y” precision and recall values obtained by each one of the feature selection methods are shown in Table 3 and Table 4, respectively. The best “Y” class precision (0.802) was achieved using backward elimination with an SMO classifier. The

⁵ <http://www-nlpir.nist.gov/projects/duc/data.html>.

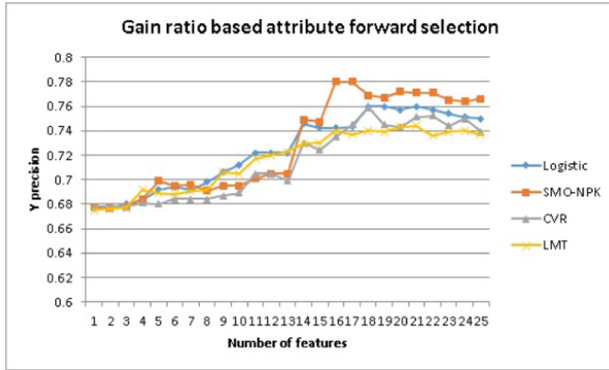


Fig. 3. Attribute selection using Gain Ratio.

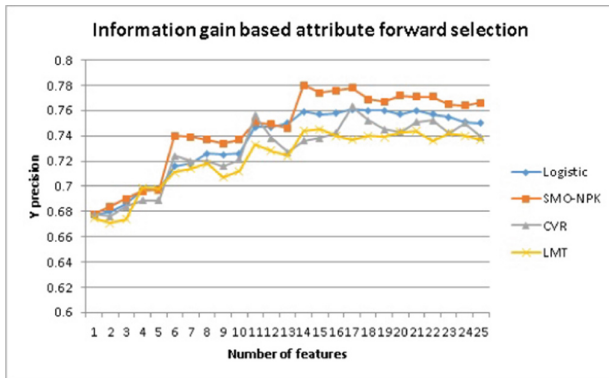


Fig. 4. Attribute selection using Info Gain.

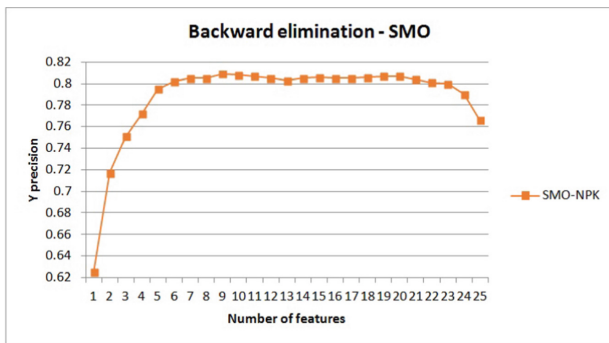


Fig. 5. Backward Elimination using “Y” precision

Table 3. “Y” class **precision** values obtained with various feature selection methods.

Method	LR	SMO-NPK	CVR	LMT
All (25) features)	0.750	0.766	0.739	0.737
Gain ratio	0.760	0.780	0.752	0.744
Info gain	0.761	0.780	0.763	0.745
Backward elimination	–	0.802	–	–

Table 4. “Y” class **recall** values obtained with various feature selection methods.

Method	LR	SMO-NPK	CVR	LMT
All (25) features)	0.754	0.741	0.761	0.771
Gain ratio	0.761	0.728	0.759	0.768
Info gain	0.767	0.735	0.759	0.771
Backward elimination	–	0.720	–	–

Table 5. “Y” class **F 0.5 measure** values obtained with various feature selection methods.

Method	LR	SMO-NPK	CVR	LMT
All (25) features)	0.751	0.761	0.743	0.744
Gain ratio	0.760	0.769	0.753	0.749
Info gain	0.762	0.771	0.762	0.750
Backward elimination	–	0.784	–	–

following six features were finally selected: *parent*, *punct_out*, *POS_before*, *POS_after*, *POS_first*, and *POS_last*. With the exception of *punct_out*, all these features are based on POS tags of specific words in the subtree and in the main sentence. Different number of features selected by each one of the feature selection methods. According to forward selection with the Gain Ratio and the Info Gain measures, “dependency” is the most important feature having the highest Gain Ratio and Info Gain values, which implies that the subtree’s dependency relation to the main sentence should be the leading criterion for choosing removal candidates in a dependency tree. However, SMO-NPK provided a better “Y” precision (0.802) than all other classifiers, without this feature and with the smallest number of features (six), when Backward Elimination was used (Table 5).

3.3 Test Data

For human evaluation, we use the first 200 sentences from a dataset available online and used in [7]. The compression ratio is defined by the number of characters in compressed sentences divided by the number of characters in the original sentences. We applied our method to each sentence several times repeatedly, with the first run processing the original sentence from the dataset and each subsequent run applied to the sentence

compressed by the preceding run. As expected, each subsequent run obtained a higher compression ratio that converged to some value after four iterations. The compression ratios (CR) for iterations 1, 2 and 4 as well as for the LSTM-based method from [7] are shown in Table 6.

3.4 Human Evaluation

We evaluated LSTM and the first, second, and fourth runs of our method in an experiment with human raters. The raters were eight graduate students not involved otherwise in this research project. The raters were asked to rate the readability and the informativeness of generated compressions. Readability measures the grammatical correctness, comprehensibility, and fluency of the output whereas informativeness measures the amount of important content preserved in the compression. Every sentence-compression pair was rated by two raters who were asked to select a score on a five-point scale. The results are shown in Table 6. The results show that our mean compression ratio is much higher than in LSTM (resulting in higher informativeness) whereas the average readability of our compressions is lower by 10% only (p value < 0.0001 for readability between our first run and LSTM). Taking into account that our model was trained on a very small dataset of only 100 sentences vs. two million in [7], our initial results are quite encouraging. Lower readability compared to LSTM may be explained by a very small training dataset and a relatively low precision of the classification model (80% only), meaning that in 20% of cases we may remove a wrong subtree and produce a grammatically incorrect sentence.

Table 6. Readability, informativeness and compression ratio of first, second and fourth iterations of our compression method compared to the LSTM.

Method	Readability	Information	CR
Run1	4.06	3.47	0.63
Run2	3.72	2.99	0.55
Run4	3.44	2.77	0.52
LSTM	4.49	3.32	0.39

4 Conclusions

We implemented a sentence compression tool that uses the subtrees of the dependency parse trees of the original sentences as potential removal candidates and calculates 25 predictive features for each subtree. To train a supervised learning algorithm we created a small dataset based on subtrees labeled by human annotators. We experimented with several classification algorithms and found that SMO (Normalized Poly Kernel) performed best with our dataset requiring only 6 out of 25 features (mostly based on POS tags) to classify the removal candidates. Subsequently, we utilized the SMO classifier

for choosing subtrees to be removed by our sentence compression pipeline and used it to compress 200 sentences from a separate benchmark dataset. The results of the human evaluation indicate that using a very small training dataset of only 100 sentences, our compressions are only slightly less readable than the compressions produced by the LSTM algorithm, which was trained on two million sentence-compression instances. Also, our pipeline can be easily adapted to other languages given a sentence splitter, tokenizer, dependency parser, and POS tagger on those languages.

References

1. Berg-Kirkpatrick, T., Gillick, D., Klein, D.: Jointly learning to extract and compress. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, vol. 1, pp. 481–490. Association for Computational Linguistics (2011)
2. Clarke, J., Lapata, M.: Global inference for sentence compression: an integer linear programming approach. *J. Artif. Intell. Res.* **31**, 399–429 (2008)
3. Cohn, T., Lapata, M.: Large margin synchronous generation and its application to sentence compression. In: EMNLP-CoNLL, pp. 73–82 (2007)
4. Cohn, T., Lapata, M.: Sentence compression beyond word deletion. In: Proceedings of the 22nd International Conference on Computational Linguistics, vol. 1, pp. 137–144. Association for Computational Linguistics (2008)
5. Colmenares, C.A., Litvak, M., Mantrach, A., Silvestri, F.: HEADS: headline generation as sequence prediction using an abstract feature-rich space. In: North American Chapter of the Association for Computational Linguistics - Human Language Technologies (NAACL HLT 2015), pp. 133–142 (2015)
6. Corston-Oliver, S.: Text compaction for display on very small screens. In: Proceedings of the NAACL Workshop on Automatic Summarization, pp. 89–98. Citeseer (2001)
7. Filippova, K., Alfonseca, E., Colmenares, C.A., Kaiser, L., Vinyals, O.: Sentence compression by deletion with LSTMs. In: EMNLP, pp. 360–368 (2015)
8. Filippova, K., Strube, M.: Dependency tree based sentence compression. In: Proceedings of the Fifth International Natural Language Generation Conference, pp. 25–32. Association for Computational Linguistics (2008)
9. Frank, E., Hall, M.A., Witten, I.H.: The WEKA workbench. online appendix for “data mining: Practical machine learning tools and techniques”, 4th edn. Morgan Kaufmann (2016)
10. Galley, M., McKeown, K.: Lexicalized markov grammars for sentence compression. In: HLT-NAACL, pp. 180–187 (2007)
11. Jing, H.: Sentence reduction for automatic text summarization. In: Proceedings of the Sixth Conference on Applied Natural Language Processing, pp. 310–315. Association for Computational Linguistics (2000)
12. Knight, K., Marcu, D.: Statistics-based summarization-step one: sentence compression. In: AAAI/IAAI, vol. 2000, pp. 703–710 (2000)
13. Knight, K., Marcu, D.: Summarization beyond sentence extraction: a probabilistic approach to sentence compression. *Artif. Intell.* **139**(1), 91–107 (2002)
14. Lin, C.Y.: Improving summarization performance by sentence compression: a pilot study. In: Proceedings of the Sixth International Workshop on Information Retrieval with Asian Languages, vol. 11, pp. 1–8. Association for Computational Linguistics (2003)
15. Manning, C.D., Surdeanu, M., Bauer, J., Finkel, J.R., Bethard, S., McClosky, D.: The stanford corenlp natural language processing toolkit. In: ACL (System Demonstrations), pp. 55–60 (2014)

16. McDonald, R.T.: Discriminative sentence compression with soft syntactic evidence. In: EACL (2006)
17. Sidhaye, P., Cheung, J.C.K.: Indicative tweet generation: an extractive summarization problem? In: EMNLP, pp. 138–147 (2015)
18. Vandeghinste, V., Pan, Y.: Sentence compression for automated subtitling: a hybrid approach. In: Proceedings of the ACL workshop on Text Summarization, pp. 89–95 (2004)

Text Categorization and Clustering



Multilabel Text Classification of Unbalanced Datasets: Two-Pass NNMF

Gabriella Skitalinskaya^(✉) and John Cardiff

Institute of Technology Tallaght, Dublin, Ireland
gabriellasky@icloud.com, john.cardiff@itt dublin.ie

Abstract. The natural distribution of textual data used in text classification is often imbalanced. Categories with fewer examples are under-represented and their classifiers trained on the datasets transformed to bag-of-words representations or basic topic modeling transformations often perform far below a satisfactory level. We tackle this problem using a two-pass non-negative matrix factorization algorithm. This approach finds topics for each category independently allowing to better define topics for underrepresented categories. The results are analyzed from multiple goal perspectives - H-loss, accuracy, F-measure, precision, and recall, from the micro, macro and example-based aspect since each is appropriate in different situations. Through experimental validation, it is shown that the two-pass matrix factorization improves classification results achieved using bag-of-words representations.

Keywords: Topic modeling · Matrix decomposition · Multi-label text classification

1 Introduction

1.1 Problem Setting

Multi-label classification is a predictive data mining problem which is applicable to a wide variety multiple real-world problems, including the automatic labeling of many resources such as texts, images, music, and video [1–5].

One of the most popular problems in this domain is text categorization (text classification), organizing text documents into several not mutually exclusive categories. The algorithms used for multi-label classification can be grouped into two classes: discriminative algorithms and generative modeling algorithms. The discriminative algorithms extend single-label algorithms so they can handle multi-label data. Excellent reviews with comparisons of discriminative algorithms are presented in [6, 7] The generative modeling algorithms model multi-label collections via the Bayes rule. According to [8–10] supervised topic models have become one of the leading generative modeling algorithms. Both classes of algorithms have their own disadvantages. The discriminative algorithm is often prone to over-fitting and highly skewed datasets, while the generative modeling

algorithm may ignore some obvious observed features. Combining two classes of algorithms allows to pursue more robust algorithms.

Instead of applying multi-label classification algorithms directly to the bag-of-words representations of document collections, in this paper we propose a two pass matrix decomposition approach based on non-negative matrix factorization (NNMF), which captures topics in a corpus of documents. The algorithm was introduced in [11] in the context of dynamic topic modeling. The two-pass matrix decomposition approach is an unsupervised technique for topic modeling, that can automatically identify topics for each category/label independently, thus, it is able to identify topics even from underrepresented categories. Representing texts according to their topic distributions is more compact than bag-of-words representation and can be processed faster than raw text in subsequent automated processes.

By using the transformed topic mixture proportions as a new representation of documents, we obtain an unsupervised dimensionality reduction algorithm that uncovers the latent structure in a document collection while preserving predictive power for the problem of classification. We demonstrate the proposed approaches applicability by analyzing news articles(Reuters) and scientific article abstracts(BibTex).

The paper is organized as follows. Section 2 describes the proposed two-pass algorithm. Measures for evaluation of topic models and classification are described in Sect. 3. Section 4 provides information on the dataset used for topic modeling. Information on the experimental setup, base classifiers parameters and results of the application of the proposed algorithm to the dataset are presented in Sect. 5. Section 6 provides a comparison of results obtained by different classifiers using data represented as topic distributions obtained by the two-pass approach with the bag-of-words data representations. Finally, Sect. 7 presents our conclusions.

1.2 Related Work

Two main approaches for solving multi-label classification problems can be identified: problem transformation methods and algorithm adaptation methods. The former transforms the multi-label problem into a single-label multi-class problem that is solved with single-label classification algorithms, e.g., binary relevance [12], classifier chains [13], random k-labelsets [14], and conditional dependency network [15], whereas the latter consists of extending a single-label algorithm so it can handle multi-label data, e.g., rank support vector machines (SVMs) [16], multi-label C4.5 [17], multi-label k nearest neighbors [18], multilabel neural networks [19], CLR [4], HOMER [20] and ECC [21].

Learning from imbalanced data is a problem which arises in many real-world datasets. Much progress has been made in developing learning algorithms dealing with imbalance based on algorithmic adaptations [22, 23], the use of ensembles [24] and resampling techniques [25–27]. One of the most deeply studied approaches lately is dealing with imbalance using resampling methods. Among the existing resampling techniques, those based on the creation of new samples

(oversampling) have shown to work better than others [28]. The new samples can be clones of existent ones, or be synthetically produced as in MLSMOTE (MultiLabel Synthetic Minority Over-sampling Technique) [29]. Multilabel oversampling algorithms based on the cloning approach proposed in [25, 26] demonstrate the approaches capability to improve classification results. In [27] a new multi-label learning approach named cross-coupling aggregation (COCOA) is proposed which is aimed at leveraging the exploitation of label correlations as well as the exploration of class-imbalance.

We have selected the following well-known and widely used methods from the literature for our benchmark comparison: Binary Relevance, Classifier Chains, ML-kNN, RAkEL and propose our own approach to dealing with unbalanced datasets.

2 Approach

2.1 Background

Non-negative matrix factorization (NNMF) is a matrix decomposition approach which decomposes a non-negative matrix into two low-rank non-negative matrices [30]. The main difference between NNMF and other classical matrix decomposition methods relies on the non-negativity constraints imposed on the model. These constraints tend to lead to a parts-based representation of the data because they allow only additive, not subtractive, combinations of data items. In this way, the factors produced by this method can be interpreted as parts of the data or, in other words, as subsets of elements that tend to occur together in sub-portions of the dataset.

Formally, non-negative matrix decomposition can be described as: $V \approx WH$ where $V \in R^{m \times n}$ is a positive data matrix with m variables and n objects, $W \in R^{m \times k}$ are the reduced k basis vectors or factors, and $H \in R^{k \times n}$ contains the coefficients of the linear combinations of the basis vectors needed to reconstruct the original data.

In the context of text analysis, for example, matrix V can be represented as a Document-Term matrix, where m is the number of documents and n the number features, matrix W represents a Document-Topic matrix, and matrix H - the Topic-Term matrix.

2.2 Two-Pass Topic Modeling Algorithm

First of all, for the purposes of the present paper the following definitions are used. The entire collection is divided into subsets of documents each containing a specific label, hereinafter “label subset”. The label subsets may overlap, since each document may have several labels. Each document may reflect one or more topics. Each topic is represented by its top-terms. Top-terms are terms that have the highest frequency (on average) in those documents that contain the topic. The number of top-terms for all topics, regardless of the category label, is the

same and is assigned by the user (for example, 5, 10, 20, etc.). When applying matrix decompositions to each label subset the user must specify the number of topics. One of the quality measures that allows us to choose the best number of topics is the so-called coherence measure [31–34].

When applying topic modeling to the entire collection, the algorithms prove to be insensitive to the topics reflected only by a small fraction of the documents, which is the typical for multi-label classification tasks. The main hypothesis of our approach is that independently modeling topics for subsets of documents for each label and subsequently aggregating the found topics into one matrix allows capturing the underrepresented topics in the collection. In the case of multi-label classification, due to the fact that subsets of documents for each label may overlap, reapplication of the matrix decomposition to the aggregated matrix allows to combine topics that are common for several labels. This reapplication of the matrix decomposition reduces the dimensionality of the data, which allows to reduce computation costs.

The approach is represented by the following algorithm:

First pass. NNMF is applied to each label subset. As a result, for each label a set of k topics is obtained, where k is defined by the user. Topics are described by a user-specified number of top-terms t and a set of all related documents.

Data Transformation. Using the topic models obtained after the first pass we construct a new compressed representation, looking through the rows of each Topic-Term matrix of each label topic model. Each row contains weights of all the terms of a particular topic of the label topic model under consideration. We construct the new Topic-Term matrix with two subsequent procedures:

- (1) In each topic from each label topic model, the top- t terms are taken from the appropriate topic-term matrix, all weights for the remaining terms are set to 0.
- (2) The obtained vectors for all label topic models are combined into one matrix.

Second pass. NNMF is re-applied to the transformed data, outputting a set of more general topics, each of which has a set of label topics associated with it. By applying matrix decompositions in this step, we identify k' general topics that potentially include topics from several labels. The number of general topics k' to be found in this step is specified by the user.

The matrix has the size $m \times n$, where m is the total number of topics in all label models, and n is the subset of the terms remaining after the data transformation. By using only the top- t terms in each topic we include only the terms that were important in any label and exclude the terms that never figured in any label topic.

3 Quality of Classification

3.1 Measures for Evaluation of Topic Modeling

Coherence measures evaluate the interpretability of the automatically generated topics and find the best number of topics. The higher the coherence score, the

better the topic model. The most widely used coherence measures to determine the optimal number of topics in each time window and the optimal number of dynamic topics, such as *UCI* [31], *NPMI* [32], *C_v* [33]. But according to [34] the TC-W2V [34] measure outperforms them.

The TC-W2V score uses the widely known word2vec tool [35] to create term vectors. In this paper we have used the Skip-gram algorithm, which predicts context words based on the current word, for estimating word representations in a vector space. The coherence of a topic represented by its t -ranked terms is determined by the mean pairwise cosine similarity between t corresponding vector-terms in the word2vec space:

$$\text{coh}(t_h) = \frac{1}{\binom{t}{2}} \sum_{j=2}^t \sum_{i=1}^{j-1} \cos(wv_i, wv_j).$$

A general evaluation of the coherence of a topic model T , consisting of k topics, is determined by the mean of individual topic coherence scores:

$$\text{coh}(T) = \frac{1}{k} \sum_{h=1}^k \text{coh}(t_h).$$

3.2 Measures for Evaluation of Classification

Performance evaluation for multi-label learning systems differs from that of classical single-label learning systems. For example, a prediction could be partially correct (some of the labels are correctly predicted), fully correct (all label predictions are correct), or fully incorrect (predictions for all labels are wrong). It is essential to include multiple and contrasting measures because of nature of the multi-label classification setting.

Metrics to evaluate bipartitions can be classified into two groups: label-based and example-based. The example-based evaluation measures are based on the average differences of the actual and the predicted sets of labels over all examples of the evaluation dataset. The label-based evaluation measures, on the other hand, assess the predictive performance for each label separately and then average the performance over all labels.

Two different label-based approaches can be used: macro and micro. Micro averaged scores give equal weight to every example and tend to be dominated by the performance in most common categories. Macro averaged scores give equal weight to every category, regardless of its frequency and is more influenced by the performance on rare categories. The macro approach is used when the system is required to perform consistently across all classes regardless of the frequency of the class (i.e., in problems where distribution of training samples across categories is skewed), whereas the micro approach may be better if the density of the class is important.

In our experiments, we used five example-based evaluation measures (Hamming loss, accuracy, precision, recall, F1 score) and six label-based evaluation

measures (micro-precision, micro-recall, micro-F1, macro-precision, macro-recall and macro-F1).

4 Dataset Description

Since we are interested in evaluating the strengths and weaknesses of our approach for different classification algorithms in a multi-label text classification context, we decided to use datasets with diverse characteristics. In our experiments to ensure comparability with other works we used two popular benchmark datasets (Reuters, BibTex) with textual data for evaluation and comparison. These datasets come pre-divided into training and testing parts: thus, in the experiments, we use them in their original format.

Reuters-21578 is one of the most widely used benchmarking collection for text categorization problems [36]. The corpus consists of news articles that appeared in the Reuters newswire in 1987. The BibTex dataset¹ is a large collection of scientific article abstracts tagged by users using 159 tags.

The obtained datasets have varying feature to label ratios and cardinality.

The key statistics for the mentioned datasets are presented in Table 1. Every document from the dataset collections went through the following preprocessing procedures:

- removal of stopwords
- removal of short words (less than 3 characters)
- lemmatization.

Also TF-IDF term weighting and document length normalization is applied to the Document-Term matrices for each label subset.

Table 1. Dataset key characteristics

Dataset	Labels	Training	Test	Features	Cardinality	Average num. of words per document
Reuters	108(55)	7713	2987	8859	1.22	162
BibTex	159	4880	2515	1836	2.40	60

5 Experiments

5.1 Experimental Setup

Word2vec models and non-negative matrix factorization has been carried out using the gensim² and sklearn³ Python libraries. The comparison of the multi-label learning methods was performed using the implementations in the following library: scikit-multilearn. Scikit-multilearn is a BSD-licensed library for multi-label classification that is built on top of the well-known scikit-learn ecosystem.

¹ The dataset can be downloaded at: <http://mulan.sourceforge.net/datasets.html>.

² <https://radimrehurek.com/gensim/>.

³ <http://scikit-learn.org/>.

5.2 Base Classifiers

To ensure comparability with other works we used the same parameter values recommended by the authors of the following papers [6, 13] or by the authors of other relevant publications.

The methods used in the mentioned papers use SVMs as base classifiers for solving the partial binary classification problems in all problem transformation methods and the ensemble methods.

In particular, [6] used the implementation based on libsvm for training SVMs with a linear basis, but in [6] the authors trained SVMs with a radial basis kernel for all problem transformation methods and RAKEL. The kernel parameter gamma and the penalty C, for each combination of dataset and method, are determined by using 10-fold cross validation only on training sets. As proposed by the authors the values $2^{-15}, 2^{-13}, \dots, 2^1, 2^3$ were considered for gamma and $2^{-5}, 2^{-3}, \dots, 2^{13}, 2^{15}$ for the penalty C.

The number of neighbors in the ML-kNN method for each dataset is determined from the values 6 to 20 with step 2.

The number of models in RAKEL is set to $\min(2Q, 100)$, where Q is the number of labels for all datasets, the size of the label-sets k is set to half the number of labels ($Q/2$) [14]. The ensemble iterations (where relevant) are set to $m = 50$.

The best parameters are determined for every method on each dataset.

5.3 Topic Modeling

To find the optimal number of topics for each label and the optimal number of general topics for each dataset using the two pass algorithm a user-specified number of top-terms used to determine the coherence of obtained topics is needed.

Due to the fact the average number of words per document in the Reuters dataset is quite high, the top-terms parameter has been set to $n = 10$ words. For the BibTex dataset, where the average number of words per document is significantly lower, the number of top-terms chosen for finding the optimal number of topics per label has been set to $n = 5$.

Depending on the number of top-terms after the first pass of the proposed approach we obtain an aggregated Topic-Term matrix for each dataset, which can be used for reducing the dimensionality of the initial dataset. Finding topics for each label independently results in overlapping topics for two or more labels. During the second pass of the proposed approach such topics are combined into more general topics, reducing the dimensionality of the data even further. Thus, for the Reuters dataset we have reduced the number of initial features from 8859 to 100, and from 1836 to 280 for the BibTex dataset. The number of features after each pass of the proposed algorithm is shown in Table 2.

Table 2. Number of features after each step of the two-pass approach

	Reuters	BibTex
Initial Num. of Features	8859	1836
First Pass.		
<i>Num. of Aggregated Topics</i>	323	1088
Second Pass.		
<i>Num. of General Topics</i>	100	280

After obtaining the general topics vectors found in the document collection, the training set is transformed using non-negative matrix factorization, solving the following problem: Given a non-negative matrix, find non-negative matrix factors W and H such that: $X \approx WH$.

We apply NNMF to the initial Document-Term matrix obtained for the training set using the precomputed General Topic- Term matrix found by the two-pass approach as H . As output we get W - the Document-General Topic matrix, which will be used as input for the classification task.

6 Classification Results

In this paper the classifiers are applied to data representations obtained after the two-pass NNMF algorithm and compared to the baseline. As the baseline we have chosen classifiers trained on bag-of-words data representations (BL1 - [13], BL2 - [6]). Classifiers are evaluated by their performance when applied to the test set from the corresponding dataset.

Results in terms of H-Loss and example-based Accuracy, Precision, Recall and F-measure are shown in Table 3 with the appropriate baseline values where possible. Since in multi-label classification different evaluation measures are appropriate in different tasks it is expected that a method may not outperform others in all measures. Table 4 gives the precision, recall and F1 scores using micro averaging, while Table 5 gives the corresponding values obtained by macro averaging along with the appropriate baseline values where possible.

As for the Reuters dataset, one can see a significant increase in the values of H-loss and example-based measures for all methods when using the two-pass NNMF approach. RAKEL performs best according to the four example-based evaluation measures, but BR performs best according to H-loss.

As for the BibTex dataset, the proposed approach performs better than the baseline only when using ML-kNN classifier according to all the used evaluation measures except for micro-precision. Though when comparing the evaluation measures obtained by ML-kNN with values obtained by other measures, it can be seen that the latter perform better.

It can be seen that the overall performance achieved on the Reuters dataset is higher than on the BibTex dataset. This could be explained by the fact that

the texts in the Reuters dataset are longer and more suitable for topic modeling. Learning topics from short texts is considered to be a challenging problem due to the severe sparsity of Document-Term data matrix, since the texts in BibTex dataset are very short(60 words per document on average), the obtained topics may be of low quality.

Table 3. Comparison of classification results using BoW and two-pass NNMF input transformation

Reuters								
	BR		CC		ML-KNN		Rakel	
	<i>BL1</i>	<i>2-Pass NNMF</i>	<i>BL1</i>	<i>2-Pass NNMF</i>	<i>BL1</i>	<i>2-Pass NNMF</i>	<i>BL1</i>	<i>2-Pass NNMF</i>
H-Loss	0.011	0.008	0.011	0.009	–	0.009	0.011	0.009
Accuracy	0.319	0.727	0.387	0.753	–	0.728	0.337	0.785
Precision	–	0.807	–	0.820	–	0.792	–	0.855
Recall	–	0.811	–	0.811	–	0.781	–	0.841
F1	0.222	0.799	0.250	0.808	–	0.779	0.233	0.840
BibTex								
	BR		CC		ML-KNN		Rakel	
	<i>BL2</i>	<i>2-Pass NNMF</i>	<i>BL2</i>	<i>2-Pass NNMF</i>	<i>BL2</i>	<i>2-Pass NNMF</i>	<i>BL2</i>	<i>2-Pass NNMF</i>
H-Loss	0.012	0.018	0.012	0.023	0.014	0.014	–	0.017
Accuracy	0.194	0.112	0.202	0.101	0.056	0.080	–	0.126
Precision	0.515	0.371	0.508	0.320	0.254	0.277	–	0.372
Recall	0.373	0.362	0.378	0.369	0.132	0.170	–	0.299
F1	0.433	0.344	0.434	0.316	0.174	0.193	–	0.304

Table 4. Comparing classification using BoW with two-pass NNMF input transformation micro P,R,F1

Reuters								
	BR		CC		ML-KNN		Rakel	
	<i>BL1</i>	<i>2-Pass NNMF</i>	<i>BL1</i>	<i>2-Pass NNMF</i>	<i>BL1</i>	<i>2-Pass NNMF</i>	<i>BL1</i>	<i>2-Pass NNMF</i>
Precision		0.863		0.821		0.835		0.828
Recall		0.751		0.745		0.711		0.775
F1		0.803		0.781		0.768		0.800
BibTex								
	BR		CC		ML-KNN		Rakel	
	<i>BL2</i>	<i>2-Pass NNMF</i>	<i>BL2</i>	<i>2-Pass NNMF</i>	<i>BL2</i>	<i>2-Pass NNMF</i>	<i>BL2</i>	<i>2-Pass NNMF</i>
Precision	0.753	0.39	0.744	0.291	0.819	0.585	–	0.425
Recall	0.328	0.328	0.335	0.338	0.118	0.147	–	0.260
F1	0.457	0.357	0.462	0.313	0.206	0.235	–	0.323

Table 5. Comparing classification using BoW with two-pass NNMF input transformation macro P,R,F1

Reuters								
	BR		CC		ML-KNN		Rakel	
	<i>BL1</i>	<i>2-Pass NNMF</i>	<i>BL1</i>	<i>2-Pass NNMF</i>	<i>BL1</i>	<i>2-Pass NNMF</i>	<i>BL1</i>	<i>2-Pass NNMF</i>
Precision	–	0.667	–	0.550	–	0.629	–	0.594
Recall	–	0.416	–	0.368	–	0.346	–	0.434
F1	–	0.474	–	0.420	–	0.416	–	0.477
BibTex								
	BR		CC		ML-KNN		Rakel	
	<i>BL2</i>	<i>2-Pass NNMF</i>	<i>BL2</i>	<i>2-Pass NNMF</i>	<i>BL2</i>	<i>2-Pass NNMF</i>	<i>BL2</i>	<i>2-Pass NNMF</i>
Precision	0.528	0.274	0.539	0.236	0.192	0.240	–	0.265
Recall	0.250	0.243	0.257	0.252	0.049	0.071	–	0.169
F1	0.307	0.229	0.316	0.222	0.065	0.096	–	0.195

7 Conclusions

In this paper, we present an algorithm based on two-pass NNMF for data transformation to improve classification results for multi-label learning with unbalanced classes. The proposed approach is able to identify topics even from under-represented categories. We evaluate the most popular methods for multi-label learning using a wide range of evaluation measures on two widely used benchmark datasets containing textual data. We compare our results to those obtained for bag-of-words representations. Our proposed topic based classifier system is shown to be competitive with existing text classification techniques.

Through experimental validation, it is shown that representing texts according to their topic distributions using the proposed two-pass approach improves classification results achieved using bag-of-words representations for longer texts such as news articles. As for shorter texts, such as abstracts for scientific articles, there is no significant increase in performance, but the achieved results are comparable to those obtained for bag-of-words data representations. This could be explained by the topic modeling nature of the proposed algorithm, as topic modeling of short texts is a problem yet to be tackled and simple matrix decomposition as NNMF used in this paper may not be able to obtain topics of high quality for short texts. Overall, representing texts according to their topic distributions is more compact than bag-of-words representation and can be processed faster than raw text in subsequent automated processes.

References

1. Schapire, R.E., Singer, Y.: BoosTexter: a boosting-based system for text categorization. *Mach. Learn.* **39**, 135–168 (2000)
2. Godbole, S., Sarawagi, S.: Discriminative methods for multi-labeled classification. In: Dai, H., Srikant, R., Zhang, C. (eds.) *PAKDD 2004*. LNCS (LNAI), vol. 3056, pp. 22–30. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24775-3_5

3. Boutell, M.R., Luo, J., Shen, X., Brown, C.M.: Learning multi-label scene classification. *Pattern Recogn.* **37**, 1757–1771 (2004)
4. Fürnkranz, J., Hüllermeier, E., Loza Mencía, E., Brinker, K.: Multilabel classification via calibrated label ranking. *Mach. Learn.* **73**, 133–153 (2008)
5. Dimou, A., Tsoumakas, G., Mezaris, V., Kompatsiaris, I., Vlahavas, I.: An empirical study of multi-label learning methods for video annotation. In: 7th International Workshop on Content-Based Multimedia Indexing, CBMI 2009, pp. 19–24 (2009)
6. Madjarov, G., Kocev, D., Gjorgjevikj, D., Džeroski, S.: An extensive experimental comparison of methods for multi-label learning. *Pattern Recogn.* **45**, 3084–3104 (2012)
7. Moyano, J.M., Gibaja, E.L., Cios, K.J., Ventura, S.: Review of ensembles of multi-label classifiers: models, experimental study and prospects. *Inf. Fusion* **44**, 33–45 (2018)
8. Ramage, D., Hall, D., Nallapati, R., Manning, C.D.: Labeled LDA: a supervised topic model for credit attribution in multi-labeled corpora. In: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, pp. 248–256 (2009)
9. Rubin, T.N., Chambers, A., Smyth, P., Steyvers, M.: Statistical topic models for multi-label document classification. *Mach. Learn.* **88**, 157–208 (2012)
10. Ma, H., Chen, E., Xu, L., Xiong, H.: Capturing correlations of multiple labels: a generative probabilistic model for multi-label learning. *Neurocomputing* **92**, 116–123 (2012)
11. Skitalinskaya, G., Alexandrov, M., Cardiff, J.: Comparison of two-pass algorithms for dynamic topic modeling based on matrix decompositions. In: Castro, F., Miranda-Jiménez, S., González-Mendoza, M. (eds.) MICAI 2017. LNCS (LNAI), vol. 10633, pp. 27–43. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-02840-4_3
12. Tsoumakas, G., Katakis, I., Overview, A.: Multi-label classification: an overview. *Int. J. Data Warehousing Mining* **3**, 1–13 (2007)
13. Read, J., Pfahringer, B., Holmes, G., Frank, E.: Classifier chains for multi-label classification. *Mach. Learn.* **85**, 333–359 (2011)
14. Tsoumakas, G., Katakis, I., Vlahavas, I.: Random k-labelsets for multilabel classification. *IEEE Trans. Knowl. Data Eng.* **23**, 1079–1089 (2011)
15. Guo, Y., Gu, S.: Multi-label classification using conditional dependency networks. In: Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence, vol. 2, pp. 1300–1305 (2011)
16. Elisseeff, A.: Kernel methods for multi-labelled classification and categorical regression problems. In: Advances in Neural Information Processing, pp. 1–18 (2002)
17. Clare, A., King, R.D.: Knowledge discovery in multi-label phenotype data. In: De Raedt, L., Siebes, A. (eds.) PKDD 2001. LNCS (LNAI), vol. 2168, pp. 42–53. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44794-6_4
18. Zhang, M.L., Zhou, Z.H.: ML-KNN: a lazy learning approach to multi-label learning. *Pattern Recogn.* **40**, 2038–2048 (2007)
19. Zhang, M.L.: ML-rbf: RBF neural networks for multi-label learning. *Neural Process. Lett.* **29**, 61–74 (2009)
20. Tsoumakas, G., Katakis, I., Vlahavas, I.: Effective and efficient multilabel classification in domains with large number of labels. In: Proceedings of ECML/PKDD 2008 Workshop on Mining Multidimensional Data (MMD 2008), pp. 30–44 (2008)
21. Read, J., Pfahringer, B., Holmes, G.: Multi-label classification using ensembles of pruned sets. In: Proceedings - IEEE International Conference on Data Mining, ICDM, pp. 995–1000 (2008)

22. He, J., Gu, H., Liu, W.: Imbalanced multi-modal multi-label learning for subcellular localization prediction of human proteins with both single and multiple sites. *PLoS ONE* **7**, e37155 (2012)
23. Li, C., Shi, G.: Improvement of learning algorithm for the multi-instance multi-label RBF neural networks trained with imbalanced samples. *J. Inf. Sci. Eng.* **29**, 765–776 (2013)
24. Tahir, M.A., Kittler, J., Bouridane, A.: Multilabel classification using heterogeneous ensemble of multi-label classifiers. *Pattern Recogn. Lett.* **33**, 513–523 (2012)
25. Charte, F., Rivera, A.J., del Jesus, M.J., Herrera, F.: Addressing imbalance in multilabel classification: measures and random resampling algorithms. *Neurocomputing* **163**, 3–16 (2015)
26. Giraldo-Forero, A.F., Jaramillo-Garzón, J.A., Ruiz-Muñoz, J.F., Castellanos-Domínguez, C.G.: Managing imbalanced data sets in multi-label problems: a case study with the SMOTE algorithm. In: Ruiz-Shulcloper, J., Sanniti di Baja, G. (eds.) *CIARP 2013. LNCS*, vol. 8258, pp. 334–342. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-41822-8_42
27. Zhang, M.L., Li, Y.K., Liu, X.Y.: Towards class-imbalance aware multi-label learning. In: *IJCAI International Joint Conference on Artificial Intelligence*, vol. 2015-January, pp. 4041–4047 (2015)
28. García, V., Sánchez, J.S., Mollineda, R.A.: On the effectiveness of preprocessing methods when dealing with different levels of class imbalance. *Knowl.-Based Syst.* **25**, 13–21 (2012)
29. Charte, F., Rivera, A.J., del Jesus, M.J., Herrera, F.: *mlsmote*: approaching imbalanced multilabel learning through synthetic instance generation. *Knowl.-Based Syst.* **89**, 385–397 (2015)
30. Lee, D.D., Seung, H.S.: Learning the parts of objects by non-negative matrix factorization. *Nature* **401**, 788–791 (1999)
31. Newman, D., Lau, J.H., Grieser, K., Baldwin, T.: Automatic evaluation of topic coherence. In: *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 100–108. Association for Computational Linguistics (2010)
32. Bouma, G.: Normalized (Pointwise) mutual information in collocation extraction. In: *Proceedings of German Society for Computational Linguistics (GSCL 2009)*, pp. 31–40 (2009)
33. Aletas, N., Stevenson, M.: Evaluating topic coherence using distributional semantics. In: *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013)-Long Papers*, pp. 13–22 (2013)
34. O’Callaghan, D., Greene, D., Carthy, J., Cunningham, P.: An analysis of the coherence of descriptors in topic modeling. *Expert Syst. Appl.* **13**, 5645–5657 (2015)
35. Mikolov, T., Corrado, G., Chen, K., Dean, J.: Efficient estimation of word representations in vector space. In: *Proceedings of the International Conference on Learning Representations (ICLR 2013)*, pp. 1–12 (2013)
36. Sebastiani, F.: Machine learning in automated text categorization. *ACM Comput. Surv.* **34**, 1–47 (2002)



Algorithmic Segmentation of Job Ads Using Textual Analysis

Benjamin Murauer¹(✉), Michael Tschuggnall¹, Günther Specht¹,
and Julia Brandl²

¹ Department of Computer Science, University of Innsbruck, Innsbruck, Austria
b.murauer@posteo.de, michael.tschuggnall@ches.pro,
guenther.specht@uibk.ac.at

² Department of Organization and Learning, University of Innsbruck,
Innsbruck, Austria
julia.brandl@uibk.ac.at

Abstract. As job ads are getting more prevalent online, an automated analysis is becoming increasingly important, especially in the field of human resource management. In this paper, we propose an approach to automatically segment job ads by predefined categories like the description of a job or the offering company, which is needed to categorize and quantify different aspects of job ads. Using a manually annotated data set, textual features are extracted for each segment type in a first step and utilized to train state-of-the-art machine learning classification methods. Subsequently, these models are used by iterative algorithms to detect the individual segments. Using several optimization techniques like detecting typical segment start phrases, comprehensive evaluations show promising results.

Keywords: Text segmentation · Job ads analysis · Structured text classification

1 Introduction

With the rise of the interconnected world wide web, the amount of data digitally transferred through various channels is increasing steadily. Especially businesses have adjusted and optimized their workflows by utilizing new possibilities of data exchange in many areas like allowing location-separated global teams or performing targeted social media marketing campaigns. Consequently, also the publication of job ads has been expanded or even completely shifted from print media to online services. With massive amounts of digitally available job ads, a crucial task in the field of human resource management (HRM) is to systematically analyze them to be able to answer questions like “What does the market want?”, “Which companies search for which person types?” or “In which domains are social skills more important?” [4, 21].

Typical study examples include attempts to detect trending key requirements demanded by employers in a specific domain, or the extraction of what is offered

by companies in return (e.g., [1,3]). Most of those studies are done by manual inspection of advertisements, which has the drawback that only a small subset of potential sources can be examined due to limitations of human resources. To conduct quantitative research, an automated textual analysis is needed. A fundamental prerequisite for making meaningful statements is thereby an automated identification of well-known segments like the description of the job tasks or requirements within ads. For example, it makes a crucial difference if the phrase “work experience” is stated within the job requirements or appears in a sentence like “You will be integrated in a team with long-term work experience”, as different conclusions can be drawn depending on the position of the phrase.

Given that a job ad depicts the employer’s offer and the future employee’s duties, the job ad content can be divided into four categories [6], whereby not necessarily all of them have to be present: the (i) **company description** introduces the offering company, who they are, what they do etc., the (ii) **job description** describes the job itself with all its components, in the (iii) **requirements** it is stated which education, technical/social skills etc. are demanded from the applicant, and finally the (iv) **offer** segment describes what the company offers, including salary and other benefits. Examples of those four categories are shown in Table 1.¹

This paper presents an approach to automatically detect those categories, if present, by applying textual analysis based on state-of-the-art machine learning techniques. It can thereby be seen as a hybrid approach that incorporates both machine learning techniques as well as specifically developed algorithms operating on top. Respectively, the two major steps are: (i) training a model in order to be able to classify each segment, and (ii) calculating the final segmentation utilizing this model. In the first step, a data set with manually assigned ground truth is used to train a model which is able to identify each segment type with high accuracy when it is given correctly separated segments as input, i.e., one of the four categories and not overlapping structures. Using this model, algorithms are designed to enable an automatic classification even if the borders are not previously known, and thus segmenting and clustering the job ad by the defined categories.

Summarizing the main contribution of this paper, we present an algorithmic text segmentation approach that is specifically tailored for job ads. Utilizing job-ad-specific knowledge, experiments reveal that algorithmic segmentation represents a valid and comparable alternative to Conditional Random Fields (CRF) [17], a state-of-the-art technique used in text segmentation and tagging.

The rest of this paper is organized as follows: Sect. 2 briefly outlines related work, and Sect. 3 explains the generation of the model in detail, which is needed as a prerequisite for the actual job ad segmentation presented in Sect. 4. The resulting basic algorithm is evaluated in Sect. 5, which is improved and reevaluated in Sect. 6. Finally, a conclusion and possible future work is discussed in Sect. 7.

¹ Note that while the example is provided in English for better understandability, the data used throughout this paper is in German.

Table 1. Examples for the different segment types (anonymized).

Category	Example 1	Example 2
Company description	In 2017, <i>CompanyX</i> not only launched a completely new product, it created and has led ever since a whole new product category. Nowadays <i>CompanyX</i> employs more than 12000 people in over 165 countries, selling over 5 billion products a year. The World of <i>CompanyX</i> provides the forum for you to use your talent and passion, to develop yourself and make an impact	About us: we look for innovation everywhere. For 130 years, we have been at the forefront of innovation, but finding solutions to the world's biggest problems has never been more important than right now. Join us today and become an essential part of the solution!
Job description	The Controller role is responsible for acting as business partner to the Sr. VP or VP of the defined area and the respective management team. Project lead and participation in the respective functional area is also part of the role. In addition, steering support processes within his/her defined area and contributing to core financial processes is expected from the position holder as well	Your responsibilities <ul style="list-style-type: none"> – Work on the software verification (component test, system test, building verification environment) – Develop automated test cases and work closely with the development team to ensure testability – Take part in the analysis of problems that occur in live operations – Travel in order to directly support our clients in the verification phase of the project
Requirements	Because our business is dynamic and advances in science and technology require new methods of production we are looking for individuals who can do the following: (i) collaborate with team members in the identification and implementation of continuous improvement initiatives and action plans. (ii) Support activities in the areas of cost containment, efficiency, productivity, energy conservation, waste minimization, operational excellence and lean practices	Your profile <ul style="list-style-type: none"> – Experience with Scrum or other iterative methodologies and experience with C# Development – Minimum of 4 years of experience designing and developing systems in a .Net environment – Experience with JavaScript, not only jQuery – Prior experience with a source control system (SVN, GIT, TFS, ...) – Good verbal and written communication skills in English and German (at least level B2)
Offer	The minimum annual salary for the position is 50,000\$ gross, whereby the effective salary is based on your qualification and experience	We offer: <ul style="list-style-type: none"> – an interesting job within the packaging & paper industry – to be part of a successful multicultural company – an empowering environment – new office building in the city center – several attractive social benefits

2 Related Work

Job advertisements have become more prevalent online, and they have shown to improve the chances of being employed [16]. Therefore, research on job ads is becoming more important in many fields, with classic examples ranging from optimizing skill sets for specific positions [1] to optimizing strategies for HR departments [11]. The information to be gained from the text differs greatly. For example, for some applications detailed features that are required by potential applicants are extracted [3], while others try to extract information to fill specific templates [7]. Also, complex frameworks have been constructed to automatically use job ads and distribute them over different channels [15], where detailed information is extracted by rule-based models, including regular expressions and other manually tailored methods. In order to improve the extraction procedures, one

goal is to detect specific topics within the text, whereby there exists no general rule of how many parts a job should be divided into. For example, [18] extract 13 different categories from the text of French job ads, including, e.g., contact information or mobility requirements, whereas [34] divide their documents into 3 more general sections (education, job title and sector). Both approaches rely on common textual features like word, character or POS-tag frequencies.

To cope with the general text and/or topic segmentation problem, a wide range of different methods is in use, often based on the research by Hearst [14], in which the lexical cohesion of terms is analyzed. Generally, approaches utilizing the lexical cohesion of segments follow either a *local* or *global* strategy. Local methods thereby process documents using fixed-sized fragments, aiming to locally detect significant changes in cohesion (e.g., [12, 14]). On the other hand, global methods try to maximize the lexical cohesion for each segment from a global perspective (e.g. [8, 32]). Due to the fact that both strategies have their pros and cons, i.e., leading to different accuracies in different situations, e.g., Simon et al. propose a graph-based hybrid approach [29]. With respect to this terminology, the presented approach in this paper can also be seen as a text segmentation problem that falls into the hybrid category, as it locally traverses text blocks using sliding windows while considering the global view. Nevertheless, in contrast to typical approaches in this field, the ad segmentation does not aim to draw borders based on topic or genre changes, but by predefined job ad categories. Moreover, segments must be attributed to those categories, i.e., only finding borders is insufficient in this case.

In terms of style markers used, approaches utilize different features and methods, e.g., Bayesian models [10], Hidden Markov Models [5], vocabulary analysis in various forms like word stem repetitions [24] or word frequency models [25]. While there exist some recent papers (e.g., [20, 26]) that include a comparison between some of the segmentation approaches on the same data sets, in general it is difficult to compare performances due to the heterogeneous problem and data types being approached. To counter this problem, the PAN workshop series² recently proposed tasks to create stylistic clusters within documents, revealing that accurate stylistic segmentation is still a difficult problem [30, 31].

With respect to automatically labelling tokens from a text document (e.g., POS tagging or Named Entity Recognition), Conditional Random Fields (CRF) have proven to be effective [19, 28]. Moreover, they have successfully been used for other sequential labeling problems, such as detecting common fields like authors, title, year or publisher from the headers and citations of research papers [22].

3 Preliminary Experiments: Training a Textual Model

To tackle the problem of automatically detecting the four predefined segment types (i) *company description*, (ii) *job description*, (iii) *requirements* and (iv) *offer* within an unseen job ad, at first a textual model is needed that can differentiate between those segments. I.e., arbitrarily given one of those segments, the model should automatically be able to predict the correct one based on textual

² <http://pan.webis.de>, visited Jan. 2018.

characteristics. In this work, two variants, i.e., models based on dictionaries and machine-learned models have been built and tested. Both variants are presented in the following and are created on the basis of a manually tagged data set consisting of approximately 1,200 manually tagged Austrian job ads, written in German language³.

3.1 Dictionary-Based Models

Due to the fact that job ads are formulated very heterogeneously, e.g., containing full sentences describing the job in detail, or consisting basically only of a job title accompanied by bullet lists, the main unit taken into account are words (including stemming and n-grams). As a first attempt, dictionaries for each category are created using the tagged data set. The prediction is then made by comparing the given segment with the dictionaries and computing similarities. Concretely, the following methods have been tested:

- (1.) *Simple Stems*: In this variant the dictionary of a segment consists of all occurring stems and their normalized weights. Given an unseen segment S , the similarity to a dictionary D is then calculated by summing up the weights in D for every occurring stem in S . By using this computation, important stems that frequently appear in a specific segment are also of higher importance with respect to the final similarity score.
- (2.) *N-Grams*: This approach is very similar to the previous one and only replaces stems by character n-grams (using $n = \{2, 3, 4, 5, 6\}$). Thus, a dictionary contains all occurring n-grams including their normalized weights, and the similarity is computed by summing up the weights of all matching n-grams.
- (3.) *Lucene*: Finally, the query model of the Java library *Lucene*⁴ has been utilized, which uses the tf-idf metric and vector space models. Thereby a document is created for each segment type by concatenating all samples, which is subsequently given to Lucene to create an appropriate index. To be specific, four different indices are created, each consisting of only one document containing all samples of the respective job ad category. Finally, the similarity of an unseen segment S to a dictionary D is then computed as follows: For each word in S , a query is formed that consists of this word only, and the similarity score for the dictionary document is retrieved from Lucene. The overall score is then computed by summing up the delivered scores of all queries (words).

3.2 Machine-Learned Models

As an alternative to the dictionary-based prediction, common machine learning techniques have been applied to build a model which can predict the segment type. The list of provided features is as follows: (i) single word stems

³ Original data provided by *textkernel*, <https://www.textkernel.com>, visited Jan. 2018.

⁴ Lucene, <https://lucene.apache.org>, visited Jan. 2018.

(*1s*), stemmed word 2-grams (*2s*) and stemmed word 3-grams (*3s*), each of them by removing or keeping stop words (*nonstop/stop*), (ii) all possible combinations of the previous features (e.g., *1s-3s-nostop* for using single stems and stemmed word 3-grams, by eliminating stop words), and (iii) character n-grams using $n = \{2, 3, 4, 5, 6\}$.

To determine the best working algorithm for this approach, several commonly used methods have been tested. Using the WEKA toolkit as a general framework [13], the following classifiers have been utilized: Naive Bayes a Bayes Network using the K2 classifier, Large Linear Classification using LibLinear, a support vector machine using LibSVM with nu-SVC classification, a k-Nearest-Neighbours classifier (kNN) using $k = 1$, PART, and a pruned C4.5 decision tree (J48). Thereby all classifiers have been used with their respective standard settings.

3.3 Model Evaluation

Both the dictionary-based as well as the machine-learned models have been evaluated using a manually tagged data set, which has been created from a subset of 1,200 samples of all Austrian job ads in 2015. All considered ads are formulated in German language, and manually segmented into the four segment types by an expert group of three persons with high experience in the human resource management field. The content of the samples thereby varies from mainly just bullet lists to fully formulated, grammatically correct sentences. Moreover, segment types may also be spread over multiple positions in the text, e.g., stating a job requirement at the beginning as well as at the end of an ad.

The data set has been divided into two halves, whereby one half has been used as the training/test corpus for the evaluation of the models, and the other half for the evaluation of the segmentation algorithm (see Sect. 5). From the 600 job ads serving for the model evaluation, 300 have been used to train the models, i.e., to build the dictionaries or to train the classifiers, respectively, and the remainder has been used for testing. Finally, the size of the dictionaries (d_s) has been parameterized (except for Lucene), choosing the dictionary to contain only the 50, 75, 100, 150 or 200 most frequent stems/n-grams, or to contain all of them. In case of the machine-learned models, the number of features has also been parameterized by the same criteria.

Table 2. Best accuracies of dictionary-based and machine-learned models in percent, given already segmented categories.

Method	d_s	cmp	Job	Offer	Req.	Avg.	Classifier	d_s	Features	cmp	Job	Offer	Req	Avg
Lucene	<i>all</i>	85.7	84.8	89.2	93.1	88.2	LibSVM	100	1s-2s-stop	94.8	94.7	97.0	97.9	96.1
5-grams	75	73.2	76.6	81.4	88.3	79.9	BayesNet	150	4-grams	92.5	92.7	97.2	96.1	94.6
6-grams	75	72.3	76.6	81.8	84.0	78.7	Naive Bayes	75	1s-3s-stop	93.0	92.1	95.5	96.4	94.2
stems	200	75.3	72.7	77.9	87.0	78.2	kNN	150	5-grams	88.2	92.1	93.2	96.8	92.6
4-grams	75	69.3	75.3	74.9	81.4	75.2	LibLinear	150	4-grams	88.0	93.1	96.3	90.8	92.0
3-grams	100	61.0	69.3	66.7	75.8	68.2	PART	100	1s-3s-stop	81.0	84.8	90.4	89.4	86.4
							J48	100	1s-3s-stop	79.1	82.2	91.2	89.6	85.5

(a) Dictionary-Based Models

(b) Machine-Learned Models

The best results for both dictionary-based and machine-learned models are shown in Table 2, where it can be seen that – while Lucene performs best for the former – most machine learning algorithms outperform it substantially. By utilizing frequencies of single stems and stemmed 2-grams, LibSVM could achieve an accuracy of 96%. In general, good accuracies are achieved for all four segment types by restricting the number of features, with the *offer* and *requirement* type having slightly better values.

4 Ad Segmentation

According to the nature of most job ads, which are given as unstructured full texts, the previously discussed classification cannot be used as is, because the borders of the different segments are not known. That is, a classifier can only predict the segment type with high accuracy if the complete and correctly separated segment is given as input. To tackle this limitation, an algorithm has been developed that is based on the constrained classifier, but allows to estimate segment borders. In the first step, a model-based probability is calculated for each text position to belong to each segment type. Subsequently, these values serve as input for the final ad segmentation, basically by locating probability peaks and applying thresholds to widen a segment.

4.1 Applying the Model

Given a full text job ad, at first a probability for each segment type and text position (token) is estimated. This is done by iteratively traversing the text using sliding windows of length w_l and step w_s , using the model to predict a vector $[p_C, p_J, p_O, p_R]$ for each window. The values in this vector correspond to the probability for the window to be of the type *company*, *job*, *offer* and *requirement*, respectively. For example, Fig. 1 shows three windows of length seven, where for each of the windows a corresponding vector is computed using the model. The window step w_s in the example is three tokens.

After processing all sliding windows, the probability vector for each token is computed by calculating the average of all predicted window vectors where the respective token appears. In the example, the first three words have been predicted by only one window, and thus the probability vectors for those words correspond to the vector of the corresponding window. On the other side, the tokens ‘*of*’, ‘*our*’, ‘*team*’ have been predicted by two windows, and thus the token vectors represent the average of those. Finally, ‘*you*’ has been predicted by three windows and is consequently calculated as the average of all three involved windows.

The overall procedure can be formalized as is stated in Algorithm 1. Given the consecutive list of tokens T of the job ad, it computes the average probability vector P_i for each token T_i by applying the model function $prob(t)$ on the respective tokens t of each window. According to the results of Sect. 3.3, machine-learned models are used for calculating the respective probabilities. Thereby, as

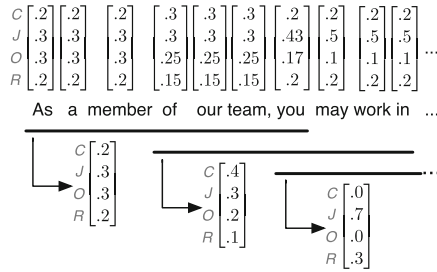


Fig. 1. Example of computing probability vectors for the categories company description (C), job description (J), offer (O) and requirements (R).

not all classifiers deliver an estimation for each class (segment type) but only the predicted class (e.g., LibSVM would only produce $[0, 0, 1, 0]$, whereas BayesNet would produce a more informative vector like $[.1, .1, .5, .3]$), the model function $prob(t)$ utilizes BayesNet instead of the slightly more accurate LibSVM classifier.

A visual example of the probability calculation for a sample job ad is depicted in Fig. 2, which shows the probabilities for each token and segment type. The text document spans on the x-axis from left to right, and the ground truth for each type is included in colored rectangles. It can be seen that each segment type is matched by the probability distribution, having peaks in the correct corresponding segment.⁵

4.2 Calculating the Final Segmentation

On the basis of the probability vectors for each token and segment type, the aim of the last step is to compute the final segmentation. As can be seen visually, probabilities like in Fig. 2 already match their corresponding segment type, i.e., having their peaks within the correct frame. To determine the whole fragments algorithmically, the basic idea is to use thresholds that widen a peak to the left and the right, respectively, until the probability falls below the thresholds. According to Algorithm 2, the procedure is as follows: (1.) Out of the unattributed segment types, choose the one with the highest peak. (2.) Starting from the peak, go to the left and attribute every token to the chosen type, until the threshold is reached or the token is already attributed to another type. (3.) Starting from the peak, do the same to the right.

During the execution, it may happen that peaks for a segment type appear within an already attributed segment. In this case, this segment type has no attribution, which also reflects the possibility that a job ad may not contain a

⁵ What can be observed in addition is that not all tokens correspond to a segment type, i.e., the text between company and job description and at the end of the ad is general and thus not attributed to a specific type.

certain segment type. In a similar way, segment types have no final attribution if an unattributed peak is found, but left and right positions are already below the given thresholds.

Algorithm 1. Probability Vector Calculation

input:
 T list of tokens
 w_l, w_s window length, window step
 $prob(t)$ computes the probability vector for the tokens t
output:
 P list of average probability vectors

```

1: for  $i$  from 1 to length( $T$ ) do  $v_i \leftarrow []$ 
2: end for

3: for  $i$  from 1 step  $w_s$  to length( $T$ ) do
4:    $t = [T_i, \dots, T_{i+w_l}]$ 
5:    $[p_C, p_J, p_O, p_R] \leftarrow prob(t)$ 
6:   for  $j$  from  $i$  to  $i + w_l$  do
7:     append  $[p_C, p_J, p_O, p_R]$  to  $v_j$ 
8:   end for
9: end for

10: for  $i$  from 1 to length( $T$ ) do
11:    $a_C \leftarrow$  average of all  $p_C$  in  $v_j$ 
12:    $a_J \leftarrow$  average of all  $p_J$  in  $v_j$ 
13:    $a_O \leftarrow$  average of all  $p_O$  in  $v_j$ 
14:    $a_R \leftarrow$  average of all  $p_R$  in  $v_j$ 
15:    $P_i \leftarrow [a_C, a_J, a_O, a_R]$ 
16: end for
    
```

Algorithm 2. Final Segmentation

input:
 P_t token probabilities for type t
 δ_l, δ_r left and right thresholds
 $maxP(T)$ returns the segment type with the highest peak
output:
 S segment type attribution per token

```

1: for  $i$  from 1 to length( $P$ ) do  $S_i \leftarrow unknown$ 
2: end for
3:  $types \leftarrow \{C, J, O, R\}$ 

4: while  $|types| > 0$  do
5:    $t \leftarrow maxP(types)$ 
6:    $p \leftarrow$  index of peak within  $P_t$ ,  $i \leftarrow p$ 
7:   while  $P_{t_i} \geq \delta_l \wedge S_i = unknown$  do
8:      $S_i \leftarrow t$ ,  $i \leftarrow i - 1$ 
9:   end while
10:   $i \leftarrow p + 1$ 
11:  while  $P_{t_i} \geq \delta_r \wedge S_i = unknown$  do
12:     $S_i \leftarrow t$ ,  $i \leftarrow i + 1$ 
13:  end while
14:   $types \leftarrow types - \{t\}$ 
15: end while
    
```

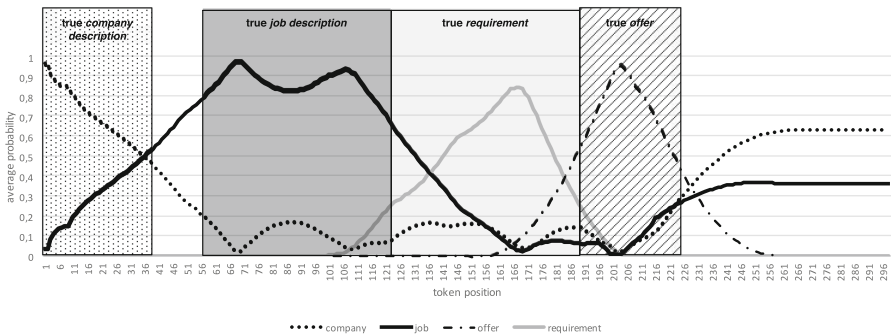


Fig. 2. Example of segment type probabilities for a sample job ad.

5 Evaluation

The presented approach has been evaluated using the remainder of the data set described in Sect. 3.3, i.e., using 600 manually tagged Austrian job ads. According to the input of the algorithms, the following parameter ranges have been evaluated: window length in tokens $w_l = \{15, 25, 50, 75\}$, window step in tokens $w_s = \{1, 2, \dots, 10\}$, left and right thresholds $\delta_l = \delta_r = \{0.3, 0.4, \dots, 0.8\}$. Typically, studies on job ads focus on analyzing the appearances of individual tokens within different segment types⁶ rather than requiring exact border positions (e.g., [2, 21]). Therefore, recall, precision and F₁-score have been used as the main metrics, which are calculated based on the comparison between the correct tokens of a segment and the predicted tokens⁷. With this method, the exact locations of segment borders are only implicitly evaluated, as only tokens - regardless of their text position - are taken into account. To also get evaluation results with respect to border positions, the commonly used text segmentation metric *WindowDiff* [23] as well as its slightly altered variant *WinPR* [27] have been utilized in addition. *WindowDiff* yields a normalized value between 0 and 1, where 0 indicates a perfect result, whereas *WinPR* computes standard recall and precision values for computed segmentations. Note that *WindowDiff* as well as *WinPR* only measure the accuracy of border positions and do not reflect the correct attribution of segments.

To put the results in perspective, two baselines have been computed: (i) the *RND-BASELINE* simply divides the job ad into four equally long segments and assigns the respective categories in a random order, and (ii) *CJRO-BASELINE* also divides the text equally into four segments, but always uses the most prominent category order for attribution, which is: *company description* – *job description* – *requirements* – *offer*.

Table 3 shows the best results per window length. Using a length of 25, i.e., 25 tokens per window, a token-based F-score of 67% could be gained, significantly outperforming the baselines. The segmentation-based F-score of *WinPR* is at 53% only, which indicates that the accuracy of finding exact border positions is substantially worse than determining relevant tokens within segments.

Table 3. Evaluation results of the basic segmentation algorithm.

w_l	w_s	δ_l	δ_r	segments				overall						
				$F_{(C)}$	$F_{(J)}$	$F_{(O)}$	$F_{(R)}$	recall	prec	F	winDiff	winP	winR	winF
25	1	0.65	0.35	65.7	73.8	54.0	71.5	65.2	70.2	67.6	35.0	52.8	53.9	53.3
15	1	0.6	0.3	62.7	75.8	46.6	67.5	62.2	70.2	66.0	35.0	50.4	50.7	50.5
50	1	0.65	0.35	67.2	61.6	62.7	70.1	70.1	61.8	65.7	37.1	50.8	47.0	48.8
75	1	0.65	0.35	66.0	56.3	64.3	65.6	70.6	57.1	63.1	38.4	48.7	43.4	45.9
CJRO-BASELINE				62.1	56.1	30.5	41.0	61.7	41.6	49.1	44.2	39.4	19.7	26.1
RND-BASELINE				25.3	21.9	21.0	22.4	29.6	20.0	23.4	44.2	39.4	19.7	26.1

⁶ E.g., if ‘abroad’ appears within the offer and/or requirement segment.

⁷ Thereby, if a segment type does not exist in the job ad but at least one token has been predicted for it, the recall/precision is set to 0, and also vice versa.

6 Improvements

Based on the previous results from Sect. 5, several techniques have been applied in order to further improve the accuracy of the approach.

1. An intrinsic characteristic of job ads is that bullet lists are used frequently, e.g., to express tasks or required skills. Moreover, it seems reasonable that segment types do not switch within bullet lists⁸. Therefore the algorithm is extended to be able to also detect bullet lists using regular expressions. If a bullet list is detected during the spanning of segments to the left and right from the probability peak, the algorithm steps over the whole bullet list and includes it in the current segment. By doing so, it is ensured that bullet lists are never split, regardless if the probability is below the left or right threshold or not.
2. When manually inspecting job ads, it can be observed that individual segment types often start with similar phrases. For example, the description of a company may start with ‘*About us:*’, ‘*Who we are:*’ or ‘*[COMPANY_NAME] is a ...*’. In order to detect such typical start sequences, additional dictionaries have been created using the training data, which store all stemmed token sequences of segment beginnings. If a newline character is found within the first three tokens, only this line is used, otherwise the first three tokens are used. Having computed a dictionary for every segment type, the segmentation algorithm from Sect. 4.2 is then altered to utilize segment beginnings. This means, that before searching for the probability peak within a segment type, it is first assessed whether a typical start sequence could be found. If yes, the matching position is used as the beginning of the segment, from where the algorithm spans now only to the right (using δ_r) and not to the left. To decide if a start position is found, common regular expressions on stems are used which match exactly, i.e., with no tolerance. On the other hand, if also the company name is involved, the extracted name of the offering company given by the data set is utilized. Because the extracted name often does not correspond exactly to the name written in the job ad, fuzzy string matching is applied⁹.

Both previously mentioned modifications have been evaluated and reveal that both can enhance the performance when applied individually¹⁰. By finally combining both improvement methods, an accuracy of 77% could be gained, as can be seen in Table 4. To compare the performance to existing segmentation approaches that do not only draw borders but are also capable of assigning classes to segments, a conditional random field (CRF) with standard parameter configuration has been trained. It was then utilized to label every token with one of the four categories, plus an artificial “*other*” label for tokens belonging to no category. It can be seen that the presented algorithmic approach outperforms the

⁸ Although there are rare cases where this is the case.

⁹ Using the Jaro-Winkler distance [9, 33] with a threshold of 0.85.

¹⁰ Due to space constraints, the individual results are omitted.

Table 4. Evaluation results including bullet list and start sequence detection.

w_l	w_s	δ_l	δ_r	Segments				Overall						
				$F_{(C)}$	$F_{(J)}$	$F_{(O)}$	$F_{(R)}$	recall	prec	F	winDiff	winP	winR	winF
50	1	0.5	0.35	69.9	79.0	72.8	86.6	79.6	75.3	77.4	28.5	63.9	57.4	60.5
75	1	0.55	0.45	69.4	78.6	72.9	88.1	79.4	75.6	77.4	29.2	62.6	57.2	59.8
25	1	0.4	0.35	69.3	73.5	68.3	73.6	70.2	74.1	72.1	29.9	61.9	59.4	60.6
CRF-BASELINE				64.1	71.0	73.1	72.1	71.9	71.6	71.5	–	–	–	–
15	1	0.4	0.3	67.8	76.5	60.3	69.7	67.3	74.0	70.5	30.2	61.0	56.3	58.5

CRF-baseline¹¹. Note that while in principle it would be possible to incorporate at least some of the algorithmic improvements also to the CRF-baseline, this task is not trivial as the CRF provides no probabilities but only labels tokens. Therefore, a different and adapted approach would have to be developed which is not addressed in this paper and left for future work.

In a final set of experiments using the hybrid approach, it has additionally been evaluated whether introducing a minimum segment length l_{min} increases the overall accuracy. By defining $l_{min} = 28$ tokens, the overall F-score could be improved slightly to 78%. On the other side, an experiment was conducted that allowed a segment to be distributed over several positions within the document¹². This was done by spanning a segment not only from one probability peak, but from two or three peaks if an individual predefined threshold was exceeded for the latter. Optimizing this threshold it has been found that the overall accuracy could not be improved, but instead is reduced by 10%. In a final experiment, it has been evaluated whether allowing different left and right thresholds for different segment types influences the segmentation accuracy, but the results indicated that this is not the case.

7 Conclusion and Future Work

In this paper, we presented an approach to automatically segment job ads into four predefined segment types. Using machine-learning based on extracted textual features, iterative algorithms operate to detect segments. Comprehensive evaluations including several optimizations using domain specific characteristics reveal that the segment types can be identified with an accuracy of 78% in average, outperforming an out-of-the-box Conditional Random Field implementation as a baseline. The proposed method may also be applied to other languages than German and evaluated, how the accuracy changes with different data sets. Moreover, it should be evaluated whether training a model which includes not only the predefined segment types, but also a specific general type (‘*other*’) can enhance the accuracy. Finally, the classifier that holds the segment type model should be tuned for an application in real-life business scenarios.

¹¹ Note that the segmentation metrics winDiff and winPR are not applicable for the CRF-baseline, as tokens are only labeled and no actual segmentation is performed.

¹² Which is the case for the *offer* segment, for example, if the salary is stated within the *job description*, while the other “offer” is formulated at a different position.

Acknowledgments. We thank textkernel for providing the job ads, as well as Gabriela Gateva and Bianca Schönherr for coding assistance.

References

1. Aken, A., Litecky, C., Ahmad, A., Nelson, J.: Mining for computing jobs. *IEEE Softw.* **27**(1), 78–85 (2010)
2. Anonymous Authors. Anonymized Title. Working Paper (2017)
3. Aureli, E., Iezzi, D.F.: Recruitment via web and information technology: a model for ranking the competences in job market. In: Proceedings of JADT 2006, pp. 79–88 (2006)
4. Jansen, B.J., et al.: Using the web to look for work: implications for online job seeking and recruiting. *Internet Res.* **15**(1), 49–66 (2005)
5. Blei, D.M., Moreno, P.J.: Topic segmentation with an aspect hidden Markov model. In: Proceedings of the 24th ACM SIGIR Conference on Research and Development in Information Retrieval, New York, NY, USA, pp. 343–348 (2001)
6. Budd, J.W., Bhavne, D.: The employment relationship. In: *The SAGE Handbook of Human Resource Management*, pp. 51–70 (2010)
7. Califf, M.E., Mooney, R.J.: Bottom-up relational learning of pattern matching rules for information extraction. *J. Mach. Learn. Res.* **4**, 177–210 (2003)
8. Choi, F.Y.: Advances in domain independent linear text segmentation. In: Proceedings of the 1st North American Chapter of the Association for Computational Linguistics Conference (NAACL), pp. 26–33. Association for Computational Linguistics, Seattle, Washington, USA, April 2000
9. Cohen, W., Ravikumar, P., Fienberg, S.: A comparison of string metrics for matching names and records. In: *KDD Workshop on Data Cleaning and Object Consolidation*, vol. 3, pp. 73–78 (2003)
10. Eisenstein, J., Barzilay, R.: Bayesian unsupervised topic segmentation. In: Proceedings of EMNLP 2008, pp. 334–343 (2008)
11. Elving, W.J.L., Westhoff, J.J.C., Meeusen, K., Schoonderbeek, J.-W.: The war for talent? The relevance of employer branding in job advertisements for becoming an employer of choice. *J. Brand Manag.* **20**(5), 355–373 (2013)
12. Ferret, O., Grau, B., Masson, N.: Thematic segmentation of texts: two methods for two kinds of texts. In: Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, vol. 1, pp. 392–396. Association for Computational Linguistics (1998)
13. Hall, M., et al.: The Weka data mining software: an update. *ACM SIGKDD Explor. Newsl.* **11**(1), 10–18 (2009)
14. Hearst, M.A.: Texttiling: segmenting text into multi-paragraph subtopic passages. *Comput. Linguist.* **23**(1), 33–64 (1997)
15. Kessler, R., Torres-Moreno, J.M., El-Bèze, M.: E-gen: automatic job offer processing system for human resources. In: Proceedings of the 6th Mexican International Conference on Artificial Intelligence, pp. 985–995 (2007)
16. Kuhn, P., Mansour, H.: Is internet job search still ineffective? *Econ. J.* **124**(581), 1213–1233 (2014)
17. Lafferty, J.D., McCallum, A., Pereira, F.C.N.: Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: Proceedings of the 18th International Conference on Machine Learning, San Francisco, USA, pp. 282–289 (2001)

18. Loth, R., Battistelli, D., Chaumartin, F.-R., de Mazancourt, H., Minel, J.-L., Vinckx, A.: Linguistic information extraction for job ads (SIRE project). In: *Adaptivity, Personalization and Fusion of Heterogeneous Information*, Paris, France, pp. 222–224 (2010)
19. McCallum, A., Li, W.: Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In: *Proceedings of HLT-NAACL*, vol. 4, pp. 188–191. ACM (2003)
20. Misra, H., Yvon, F., Jose, J.M., Cappe, O.: Text segmentation via topic modeling: an analytical study. In: *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, pp. 1553–1556. ACM (2009)
21. Müller, O., et al.: Towards a typology of business process management professionals: identifying patterns of competences through latent semantic analysis. *Enterp. Inf. Syst.* **10**(1), 50–80 (2016)
22. Peng, F., McCallum, A.: Information extraction from research papers using conditional random fields. *Inf. Process. Manag.* **42**(4), 963–979 (2006)
23. Pevzner, L., Hearst, M.A.: A critique and improvement of an evaluation metric for text segmentation. *Comput. Linguist.* **28**(1), 19–36 (2002)
24. Ponte, J.M., Croft, W.B.: Text segmentation by topic. In: Peters, C., Thanos, C. (eds.) *ECDL 1997*. LNCS, vol. 1324, pp. 113–125. Springer, Heidelberg (1997). <https://doi.org/10.1007/BFb0026725>
25. Reynar, J.C.: Statistical models for topic segmentation. In: *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pp. 357–364. ACM, Maryland, USA (1999)
26. Riedl, M., Biemann, C.: Topictiling: a text segmentation algorithm based on LDA. In: *Proceedings of ACL 2012 Student Research Workshop*, pp. 37–42. ACM (2012)
27. Scaiano, M., Inkpen, D.: Getting more from segmentation evaluation. In: *Proceedings of HLT-NAACL*, pp. 362–366. ACM, Montreal, Canada (2012)
28. Sha, F., Pereira, F.: Shallow parsing with conditional random fields. In: *Proceedings of HLT-NAACL*, pp. 134–141. ACM, Edmonton, Canada (2003)
29. Simon, A.-R., Gravier, G., Sébillot, P.: Leveraging lexical cohesion and disruption for topic segmentation. In: *International Conference on Empirical Methods in Natural Language Processing, EMNLP 2013*, pp. 1314–1324 (2013)
30. Stamatatos, E., et al.: Clustering by authorship within and across documents. In: *Working Notes Papers of the CLEF 2016 Evaluation Labs*, September 2016
31. Tschuggnall, M., et al.: Overview of the author identification task at PAN-2017: style breach detection and author clustering. In: *Working Notes Papers of the CLEF: Evaluation Labs*, p. 2017. CLEF and CEUR-WS.org, September 2017
32. Utiyama, M., Isahara, H.: A statistical model for domain-independent text segmentation. In: *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pp. 499–506 (2001)
33. Winkler, W.E.: The state of record linkage and current research problems. In *Statistical Research Division, US Census Bureau*. Citeseer (1999)
34. Zavrel, J., Berck, P., Lavrijssen, W.: Information extraction by text classification: corpus mining for features. In: *Proceedings of the Workshop Information Extraction Meets Corpus Linguistics* (2000)



Neural Network Architecture for Credibility Assessment of Textual Claims (Best Paper Award, First Place)

Nurendra Choudhary^(✉), Rajat Singh, Ishita Bindlish, and Manish Shrivastava

Language Technologies Research Centre (LTRC), Kohli Center on Intelligent Systems
(KCIS), International Institute of Information Technology, Hyderabad, India
{nurendra.choudhary, rajat.singh}@research.iiit.ac.in,
ishita.bindlish@students.iiit.ac.in, m.shrivastava@iiit.ac.in

Abstract. Text articles with false claims, especially news, have recently become aggravating for the Internet users. These articles are in wide circulation and readers face difficulty discerning fact from fiction. Previous work on credibility assessment has focused on factual analysis and linguistic features. The task's main challenge is the distinction between the features of true and false articles. In this paper, we propose a novel approach called Credibility Outcome (CREDO) which aims at scoring the credibility of an article in an open domain setting.

CREDO consists of different modules for capturing various features responsible for the credibility of an article. These features includes credibility of the article's source and author, semantic similarity between the article and related credible articles retrieved from a knowledge base, and sentiments conveyed by the article. A neural network architecture learns the contribution of each of these modules to the overall credibility of an article. Experiments on Snopes dataset reveals that CREDO outperforms the state-of-the-art approaches based on linguistic features.

Keywords: Credibility analysis · Fake news · Model ensembles

1 Introduction

“Fake news is a type of hoax or deliberate spread of misinformation, be it via the traditional news media or social media, with the intent to mislead, in order to gain financially or politically” [14].

The fake articles do not base their information on facts but use convenient, seemingly true, logical inferences to make readers trust the news. In recent times, the difference between real and fake news articles has become bleak. Verification of the information needs checking for reliable sources. Most readers find the task cumbersome and hence don't perform it. Therefore, automated systems that detect such articles are a major requirement.

N. Choudhary and R. Singh—These authors have Contributed equally to this work.

© Springer Nature Switzerland AG 2023

A. Gelbukh (Ed.): CICLing 2018, LNCS 13397, pp. 301–313, 2023.

https://doi.org/10.1007/978-3-031-23804-8_24

Correction of misinformation is difficult. Fabricated news persists because of people casually inferring based on available information about a given event. As a result, false information continues to influence opinions, beliefs and attitudes even after being debunked, unless replaced by an alternate factual explanation. The situation deteriorates when the fake articles have a financial or political agenda. The manner in which these articles shape public opinion affects the society as a whole, making it a very serious problem.

The modules of CREDO's (Credibility Outcome) include keyword extraction, document retrieval, author credibility scores, website trust scores, semantic similarity and sentiment analysis. Keyword extraction module chooses significant words in the given input article which, in the document retrieval phase, fetches documents from a knowledge base, primarily, news pages and Wikipedia. Querying the entire document gives us negative results and is not efficient, considering the long queries. Subsequent phases learn trust scores of the source websites of the retrieved articles and credibility of their authors. To ensure that our retrieved document and the given article are not dissimilar, we also compute semantic similarity. An article is a sequence of words. Semantic similarity modules need to include a method that captures this sequence of words and also learn a distance metric between them. Long Short-Term Memory (LSTM) based recurrent neural networks have proven helpful in capturing sequences [19]. Also, siamese networks have shown promising results in distance-based learning methods [5]. Hence, we use a combination of these by utilizing a bidirectional LSTM to map articles to a semantic space in conjunction with a siamese network to learn the similarity metric between them.

A factual article has more probability of being neutral [18]. Hence, a system to capture this feature is essential. To tackle the problem, we use a sentiment analysis [20] tool to evaluate the neutrality of a given article.

The rest of the paper is organized as follows. We discuss previous approaches in the field, motivating us for the task in Sect. 2. In Sect. 3, we discuss the pipeline and individually explore each module of CREDO. Section 4 describes the datasets and baselines considered for the task. Section 5 explains our experiments and their evaluation. Finally, Sect. 6 concludes the paper.

2 Related Work

The work connects a significant number of research areas, including but not limited to automatic fact checking, rumour detection, sentiment analysis, semantic similarity and credibility analysis.

[29] discusses the belief system of computers and credibility analysis' necessity. [28] discusses fact checking, its definition and motivations, posing it as a classification task. [7] proposes credibility analysis in a social media context, thereby using features that describe users' posting behaviour. Joint model based on CRF [22] employs linguistic features like assertive verbs, discourse markers among others to establish a common style of writing across fake articles and news. This approach proves effective and achieves more than 80% accuracy on snopes dataset.

CoTruth [15] uses a similar approach in employing linguistic features as a joint model based on CRF [22] but does not perform a credibility check on the author and website.

The above approaches model the writing style. However, they do not consider the available knowledge bases. The writing style also depends on the content’s authors and the website’s type. A unified model to capture the style without taking into account the information of authors and website will undoubtedly be fragile.

In semantic similarity, we use siamese networks, to compute the similarity between two sentences. [5] first introduced siamese networks in 1994 to solve the problem of signature verification. Since then, there have been attempts to understand their relevance in the context of sentence similarity. In SCQA model [9], siamese networks solve the task of community question answering and in DSSM model [12], they handle the task of website ranking. The above methods use the siamese network based on the task. Here, the task is the semantic similarity. Hence, we use LSTM models [19] to project our articles in the semantic space to learn a similarity metric between them.

Table 1. Result of keyword extraction

Article	Keywords
In May 1946, Einstein made a rare public appearance outside of Princeton, New Jersey, when he traveled to the campus of Pennsylvania’s Lincoln University, the United States’ first degree-granting black university, to take part in a ceremony conferring upon him the honorary degree of doctor of laws	[[('degree-granting black university', 8.5), ('lincoln university', 4.5), ('ceremony conferring', 4.0)]]
There are two fatal problems with the JATO story. First, anybody who understood the extreme forces involved well enough to attach a JATO unit to a car so that it would keep the car going in a straight line would probably know better than to do it in the first place	[[('extreme forces involved', 9.0), ('fatal problems', 4.0), ('straight line', 4.0)]]

3 CREDO: Methodology and Architecture

CREDO or Credibility Outcome is a neural network architecture combining information retrieval, semantic similarity, and sentiment analysis. The following subsections explain the architecture’s modules.

3.1 Keyword Extraction and Document Retrieval

Keywords provide us with most of the significant information contained in the article. Hence, keyword extraction becomes an essential initial step for document retrieval. The module creates a query using keywords of input text to retrieve relevant documents. We use Rapid Automatic Keyword Extraction (RAKE) described in [25]. RAKE involves three sub-modules -

1. **Candidate selection** extracts all potential keywords (words, phrases and concepts).
2. **Properties calculation** measures the candidate’s indication of being a keyword. For example, a likely keyword is a candidate in the title of a book.
3. **Scoring and selecting keywords** scores the candidates by either combining the properties into a formula or utilizes a machine learning technique to determine the probability of a candidate being a keyword. The final set are the keywords with a score above an empirical threshold. This assigns the keywords to an article, with their respective scores representing their importance in the article. Some examples of the extracted keywords are given in Table 1.

Keyword extraction and document retrieval module uses the keywords to query over the knowledge bases (proprietary set of indexed documents, such as Wikipedia, or a Web search engine like Google search and Bing search) using an information-retrieval system. The result of this document retrieval stage is a set of relevant documents.

Although the set of documents is relevance-ranked, the top entities are probably not credible enough. Keyword relevance is not an appropriate ranking metric for a credibility scoring system. The existence of a highly relevant and large document full of fake news is easily possible. Therefore, we rely on other modules to check document credibility.

3.2 Website and Author Trust Scores

In this module, given an article’s source, the Web of Trust API¹ initializes the credibility score of the website. Web of Trust, a website reputation and review service, provides information about the trustworthiness of a website. The information’s basis is a combination of crowd-sourced reviews and identification of networks involved in malware distribution.

With the scores from Web of Trust, we perform a logarithmic gradient descent based on the tag of the input in the dataset. i.e.

$$WTS(w_{i+1}) = \begin{cases} (i + \log(1 - t(w_i))) \times WTS(w_i), t(w_i) \leq 0.5 \\ (i + \log(1 + t(w_i))) \times WTS(w_i), t(w_i) > 0.5 \end{cases} \quad (1)$$

where $WTS(w_i)$ is the web trust score of website w for its i^{th} instance in the sample and $t(w_i)$ represents the tag of the i^{th} article.

Similarly, initial articles of an author decide his score for the present one. An initial score of 0.5 is set for each author. If the author is anonymous or irretrievable from the dataset, the probability of an author being credible and not credible is assumed to be equal and hence, the ACS score is set to 0.5. The metric for author credibility score is:

$$ACS(a_{i+1}) = \begin{cases} (i + \log(1 - t(a_i))) \times ACS(a_i), t(a_i) \leq 0.5 \\ (i + \log(1 + t(a_i))) \times ACS(a_i), t(a_i) > 0.5 \end{cases} \quad (2)$$

¹ www.mywot.com.

where $ACS(a_i)$ is the author credibility score of author a at his i^{th} article in the sample and $t(a_i)$ denotes the tag of the i^{th} article.

New training instances dynamically update these calculated scores according to their contribution to the final credibility Score of the source website and the article’s author.

3.3 Document Summarization

Only specific sections of the document demonstrate the significant relevant information in a text article. Hence, based on the keywords and TextRank algorithm [17] with BM25 ranking function [2] (implementation by Gensim²), we summarize the document by extracting the top k sentences based on the algorithm’s ranking. We choose k such that the summary size is approximately the same as that of the input article. Summarization of large input articles emphasize the most significant information, while at the same time, simplifying keyword extraction.

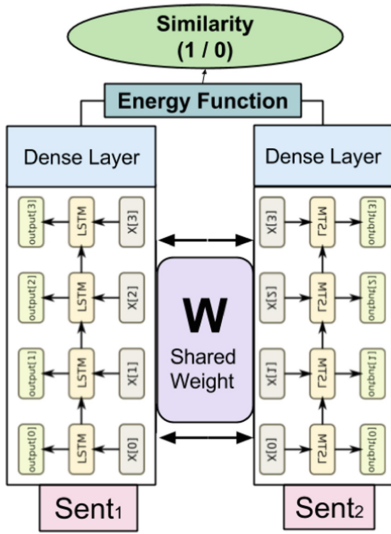


Fig. 1. Siamese architecture for semantic similarity

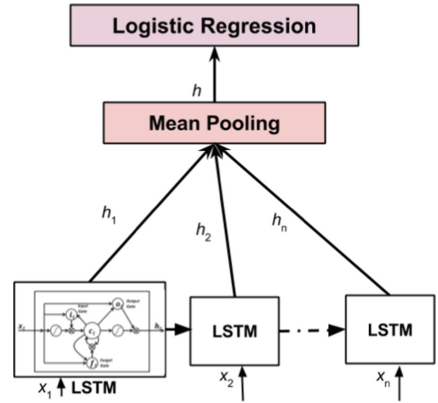


Fig. 2. LSTM RNN for sentiment classification

3.4 Semantic Similarity

Semantic similarity is of paramount importance in the scoring task. Given the summarized versions of the original input article and the documents retrieved in the previous steps, we calculate the semantic similarity score between them. This

² <https://radimrehurek.com/gensim/summarization/summariser.html>.

is necessary because only keywords influence the documents retrieval from the knowledge base. So, it is possible that the keywords match the negative meaning or the document retrieved and the original input are unrelated, apart from just some words.

For example, a web search on the query “Donald Trump is the president of India” results in titles - “How US President Donald Trump’s world-view affects India”, “US President Donald Trump to speak to PM Modi as India hopes to build on US ties” etc., which are considered good evidences by the system but are semantically different.

In this model, siamese architecture with LSTMs calculates the semantic similarity, similar to how the DSSM model [12] computes semantic similarity with primarily two differences:

- A fully connected neural network is the DSSM model’s basis, whereas, we employ an LSTM here instead. LSTMs capture sequential information data types like texts, in the sense that they capture order and history.
- The basic unit in their model constitutes character n-grams, whereas we use tokens or words. We do not need character based vectors because news articles are not susceptible to spelling errors and out of vocabulary words.

As illustrated in Fig. 1, the architecture consists of twin LSTM networks with a similarity-based energy function on the top. The LSTM networks map the article to a vector in the semantic space and the energy function learns the similarity between them. The similarity metric is learned using contrastive learning. The sentences with similar meaning are labeled +1 (positive samples) and dissimilar meaning sentences are labeled -1 (negative samples). The results of semantic similarity are explained later in Sect. 5.2.

3.5 Sentiment Analysis

Sensationalized or opinionated news articles are sentiment heavy whereas factual articles are more objective and neutral. To establish a relation between credibility and sentiment of an article, we employ a sentiment analysis tool based on LSTM (Long Short-Term Memory) networks. LSTM networks show remarkable results in learning sequential information like texts [27]; [11]. As depicted in Fig. 2, we use an LSTM model with single layer. It transforms the input sequence $\langle x_i \rangle$ to a representation sequence $\langle h_i \rangle$. Then a pooling layer averages representations h_i s over all time steps, resulting in a single representation. Finally, we train a logistic regression model on the representations whose target is the sentiment label (+1 or -1) corresponding to the input sequence. Our motivation to use an LSTM based sentiment classification model without using any manual linguistic features is to avoid any linguistic encoding throughout CREDO and rely essentially on machine intelligence. Section 5.2 describes the influence of this module over the system.

3.6 Ensemble of the Modules

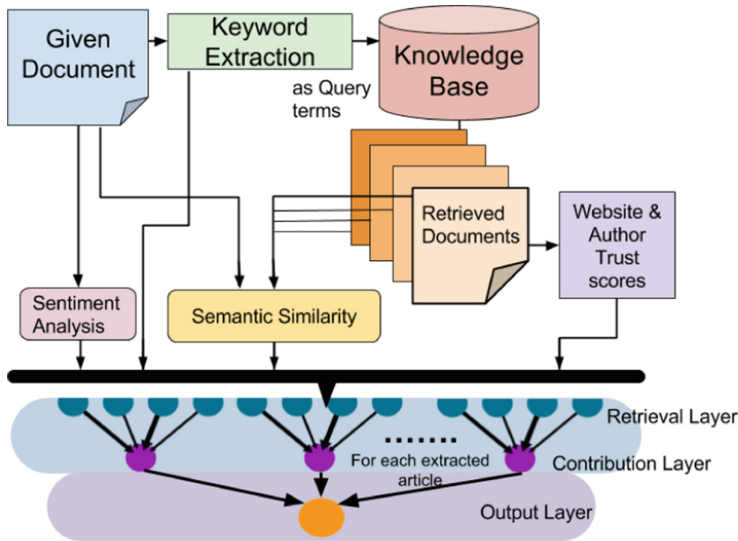


Fig. 3. Architecture of CREDO

Taking Naive Bayes assumption (or conditional independence assumption), we consider all the above features independent of each other. They together contribute to the article’s overall credibility. So, credibility contribution of each article is given by:

$$\begin{aligned}
 CC(a_r, a_i) = & (w_1 \times (\sum_{i \in k} kscore(i))) + (w_2 \times ACS + w_3 \times WTS) \\
 & + (w_4 \times NS) + ((w_5) \times SS(a_i, a_r)) \\
 \text{given that, } & w_1 + w_2 + w_3 + w_4 + w_5 = 1 \ \& \ w_1, w_2, w_3, w_4, w_5 > 0 \quad (3)
 \end{aligned}$$

where a_r, a_i are the retrieved article and given input article respectively, $CC(a_r, a_i)$ represents the credibility contribution of an article a_r with respect to the given article a_i , k represents the keywords in the article a_r $kscore(i)$ represents the keyword scores of the i^{th} keyword, ACS represents the author’s credibility score derived from Eq. 2, WTS represents the trust score of the website derived from Eq. 1, $SS(a_i, a_r)$ represents the similarity score between the input and retrieved article and NS represents neutral sentiment’s value of a_i , scaled in 0–1. w_1, w_2, w_3, w_4, w_5 are the respective weight parameters given to these measures in the overall credibility.

Since we retrieve multiple articles, we repeat the same process for every one. We assume that the credibility of all the retrieved articles, weighted with a

Table 2. Comparison of CREDO with baselines (LG+SR and CRF).

Experiment	Overall accuracy	True claims accuracy	False claims accuracy	Macro-averaged accuracy	AUC	Fake claims precision	Fake claims Recall	Fake claims F1-score
CRF	81.39	83.21	80.78	82.00	0.88	0.93	0.81	0.87
LG + SR	71.96	75.43	70.77	73.10	0.80	0.89	0.71	0.79
RBF-SVM	82.8	85.7	81.5	83.6	0.87	0.94	0.85	0.89
MLP-NN	83.3	86.2	81.2	83.7	0.83	0.92	0.84	0.88
Multi-class	56.4	59.8	55.2	57.5	0.63	0.63	0.53	0.57

function of their respective retrieval ranks, will contribute to the final credibility score for the given input article.

$$Credo(a_i) = \frac{\sum_{r \in R} (e^{1 - \frac{rank(a_r)}{n}} \times CC(a_r))}{\sum_{r \in R} e^{1 - \frac{rank(a_r)}{n}}} \quad (4)$$

where $Credo(a_i)$ is the input article’s overall credibility a_i , R denotes the set of retrieved documents, n denotes the number of documents retrieved, $rank(a_r)$ represents the retrieval rank of article a_r and $CC(a_r)$ denotes the credibility contribution of a_r given by Eq. 3. Equation 4 gives us the input article’s credibility in the range $[0, 1]$. Figure 3 details the overall architecture of CREDO.

The weights in the Eqs. 3 and 4 define the parameters for the module’s linear combination scores. Various types of classifiers learn these parameters. Section 5 explains the different trials. The weights learned assigns the credibility score to any new input text article.

4 Dataset and Baselines

This section explains the choice of Dataset for the task and also describes the construction of baselines from the previous approaches to compare with CREDO.

4.1 Dataset

The following datasets are considered for training and evaluation of CREDO.

Snopes Dataset: The dataset used for comparative analysis of the Credibility Scoring Algorithm is Snopes Dataset. Previous approaches to the problem [22], [21] use the same dataset.

Each article on Snopes verifies a single claim. The Snopes’ editors assign a manual credibility verdict to each such claim: True or False. Few of the claims have labels *Mostly True*, *Mostly False*, *Partially True* or *Partially False*. A description of the editors’ arrival to the claim accompanies the credibility verdict (e.g., collected from a Facebook post or received by email etc.) - an *Origins*

section describes the claim’s origin, and a *Description* section justifies the verdict. We consider *True* and *Mostly True* as *True* claims. Similarly, we consider *False* and *Mostly False* as *False*. The dataset has 4856 total claims with 1277 *True* claims and 3579 *False* claims.

SemEval 2016 Dataset: For evaluating the Semantic Similarity module, we adopt the standard SemEval-2016 Task1 - English Semantic Textual Similarity (STS) Dataset³. The dataset consists of sentence pairs with their similarity scores in binary.

The dataset has five different sentences’ types drawn from various sources. The dataset of *Answer-Answer* and *Question-Question* is taken from Stack Exchange Q&A Forums⁴, *Headlines* data is taken from Europe Media Monitor (EMM) [3], data of *Plagiarism* is taken from corpus of plagiarized short answers [8] and data of *Postediting* is taken from WMT quality estimation shared task [6].

4.2 Baselines

We test CREDO system against the previous approaches in the problem [21, 22] and evaluate it against the same metrics.

- **LG + SR:** This approach uses language stylistic features and source reliability to determine the credibility tag of an article using a distant supervision model [21].
- **CRF:** This approach improves upon the *LG+SR* and attempts to solve the problem using Conditional Random Fields (CRF) dependent on web-sources, articles, claims and claim credibility labels [22].

5 Experiments

We conducted an array of experiments to understand the effect of different modules on the system’s overall performance. For this, we first excluded the modules and then observed the effect of this exclusion on the system’s evaluation metrics. We also conducted experiments to understand the effect of different classifiers on the system. We calculated the evaluation metrics for different classifiers and attempted to understand the reasons for the difference in results. Semantic similarity, being a major module, was evaluated independently to improve results.

5.1 CREDO System

We train and test the CREDO system on the Snopes dataset. The experiment is conducted for binary credibility classification. The *mostly true* and *true* are considered positive labels and *mostly false* and *false* are considered negative

³ <http://alt.qcri.org/semeval2016/task1/index.php?id=data-and-tools>.

⁴ <https://archive.org/details/stackexchange>.

Table 3. Comparison between the full CREDO system and CREDO system without some modules. SS here is Semantic Similarity and SA is Sentiment Analysis

Excluding modules	Overall accuracy	True claims accuracy	False claims accuracy	Macro-averaged accuracy	AUC	Fake claims precision	Fake claims Recall	Fake claims F1-score
CREDO <i>w/o</i> ACS	80.5	82.3	81.3	81.8	0.83	0.90	0.82	0.86
CREDO <i>w/o</i> WTS	81.8	84.7	80.4	82.55	0.85	0.92	0.83	0.87
CREDO <i>w/o</i> SS	50.6	53.4	50.4	51.9	0.53	0.58	0.56	0.57
CREDO <i>w/o</i> SA	76.2	78.4	75.4	76.9	0.76	0.77	0.76	0.76
CREDO	83.3	86.2	81.2	83.7	0.83	0.92	0.84	0.88

Table 4. Correlation score in STS 2016 Task-1 of CREDO semantic similarity module

Dataset	Ans. -Ans.	Headlines	Plagiarism	Postediting	Ques.-Ques.	Mean
Baseline STS	0.41	0.54	0.70	0.83	0.04	0.51
CREDO similarity	0.54	0.71	0.73	0.82	0.61	0.68
Improvement (%)	31.7	31.5	4.3	-1.2	1425	33.3

labels. However, for our second experiment, we conduct multi-class credibility classification. For this, *true, mostly true, mostly false* and *false* are considered separate labels in a decreasing order.

Accuracy, Precision, Recall and F1-score are the evaluation metrics of this experiment. For training the weights, different classifiers were used - Support Vector Methods(SVM) with RBF kernel [1] and Neural Networks (Multi-Layer Perceptron) [24] had the best results. Evaluation metrics for multi-class SVM use methods from [10].

K-fold cross validation ($K = 5$) was used for training and testing on Snopes dataset. The evaluation metrics obtained are averaged across all the evaluation sets. The results of experiments with different classifiers are given in Table 2.

5.2 Dependency of CREDO on Its Modules

CREDO is a system based on 5 modules, but not all of them contribute equally. So, this experiment studies each modules' contribution to the overall system. For this, the effect on the performance of CREDO due to the exclusion of modules is analyzed in contrast to the original system.

Semantic similarity has to be tested independently to observe its effect on the overall performance of CREDO. The performance of semantic similarity module is tested on standard SemEval-STS 2016 English task.

5.3 Dependency of CREDO on the Classifier

The overall problem is of classification and hence the choice of classifier has to be appropriate for the data points. Quadratic Discriminant Analysis (QDA) [26], Gaussian Naive Bayes [13], Decision Trees [23], Random Forests (ensemble of Decision Trees) [4], AdaBoost Classifier [16], Support Vector Methods with

Table 5. Comparison between different classifiers.

Classifier	Overall accuracy	True claims accuracy	False claims accuracy	Macro-averaged accuracy	AUC	Fake claims precision	Fake claims Recall	Fake claims F1-score
Quadratic DA	61.6	63.5	59.5	64.5	0.67	0.63	0.58	0.60
Naive Bayes	65.8	67.1	64.3	66.7	0.68	0.71	0.65	0.69
Decision Trees	66.3	67.4	64.5	65.4	0.69	0.70	0.67	0.68
AdaBoost	70.6	72.3	69.4	71.1	0.74	0.75	0.72	0.73
Random Forest	78.2	80.1	79.8	80.8	0.81	0.82	0.80	0.81
RBF-SVM	82.8	85.7	81.5	83.6	0.87	0.94	0.85	0.89
MLP-NN	83.3	86.2	81.2	83.7	0.83	0.92	0.84	0.88

Radial Basis Function kernel (SVM-RBF) [1] and Multi Layer Perceptron Neural Network (MLP-NN) [24] were chosen for the experiments based on the variance of their application. All the evaluation metrics, tested for the original system, were tested for the classifiers as well.

5.4 Evaluation of the Experiments

Table 2 shows the comparative results of CREDO on Snopes dataset. CREDO outperforms the state-of-the-art approaches across all the metrics. It is also observed that CREDO shows improvements over other approaches, even without the help of the Web of Trust module. Hence, the combination of Web of Trust and author credibility scoring modules helps in boosting the performance.

The results for the experiment on dependency of CREDO on modules (given in Sect. 5.2) is shown in Table 3. The results show that ACS and WTS scores help the model but the improvements are insignificant compared to the enhancement in the metrics brought by the semantic similarity module. Including sentiment analysis module increments the accuracy, but it is minor compared to the improvement given by semantic similarity. At the same time, also not as insignificant as ACS and WTS scores. The results for the semantic similarity module are given in Table 4. The results show the improvements the model had over the baseline system of the organizers in terms of Pearson Correlation score between the true similarity in the dataset and the scores of the given model.

The comparative results of the experiment with different classifiers (given in Sect. 5.3) are given in Table 5. As observed, the RBF-SVM model and MLP-NN model are better than the other architectures and have similar scores. Hence, these models were chosen for the main architecture.

6 Conclusion

We have proposed a neural network based approach for credibility analysis of unstructured text articles in an open-domain setting. We use a combination of relevant document retrieval techniques with semantic similarity, sentiment analysis and source reliability of articles then reporting the credibility score of

the given input. Experiments on Snopes data demonstrate the effectiveness of our approach compared to the previous approaches to the problem.

Experiment on contribution of modules (given in Sect. 5.2) shows that the website and author reliability are helpful, but only to a limited extent. The accuracy scores had a difference of about 1%. Semantic similarity and document retrieval are the major contributors to the system. The exclusion of the semantic similarity module resulted in a slash of 32.7%. Sentiment Analysis contributed 7.1% accuracy. Hence it can be concluded that the most important module is semantic similarity, followed by sentiment analysis, and then author, website scores.

The experiments also show that the choice of the classifier plays a major role in the output. Support Vector Methods with RBF kernel and Neural Network architecture (Multilayer Perceptron) give the best results for the problem. This is because the data points are not trivially classifiable and require non-linear classification. The neural network solves it by using multiple lines to classify the dataset whereas SVM utilizes the non-linear RBF kernel to address the problem.

As part of future work, we would like to apply this system to domains like social media. Also, we would like to enhance CREDO with more handcrafted features like the writing style.

References

1. Amari, S.I., Wu, S.: Improving support vector machine classifiers by modifying kernel functions. *Neural Netw.* **12**(6), 783–789 (1999)
2. Barrios, F., López, F., Argerich, L., Wachenchauser, R.: Variations of the similarity function of textrank for automated summarization. arXiv preprint [arXiv:1602.03606](https://arxiv.org/abs/1602.03606) (2016)
3. Best, C., van der Goot, E., Blackler, K., Garcia, T., Horby, D.: Europe media monitor. Technical report EUR221 73 EN, European Commission (2005)
4. Breiman, L.: Random forests. *Mach. Learn.* **45**(1), 5–32 (2001)
5. Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., Shah, R.: Signature verification using a “Siamese” time delay neural network. In: *Advances in Neural Information Processing Systems*, pp. 737–744 (1994)
6. Callison-Burch, C., Koehn, P., Monz, C., Post, M., Soricut, R., Specia, L.: Findings of the 2012 Workshop on Statistical Machine Translation. In: *Proceedings of the Seventh Workshop on Statistical Machine Translation. WMT 2012*, pp. 10–51. Association for Computational Linguistics, Stroudsburg (2012). <http://dl.acm.org/citation.cfm?id=2393015.2393018>
7. Castillo, C., Mendoza, M., Poblete, B.: Information credibility on Twitter. In: *Proceedings of the 20th International Conference on World Wide Web*, pp. 675–684. ACM (2011)
8. Clough, P., Stevenson, M.: Developing a corpus of plagiarised short answers. *Lang. Resour. Eval.* **45**(1), 5–24 (2011)
9. Das, A., Yenala, H., Chinnakotla, M., Shrivastava, M.: Together we stand: Siamese networks for similar question retrieval. In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, pp. 378–387 (2016)

10. Godbole, S., Sarawagi, S.: Discriminative methods for multi-labeled classification. In: Dai, H., Srikant, R., Zhang, C. (eds.) PAKDD 2004. LNCS (LNAI), vol. 3056, pp. 22–30. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24775-3_5
11. Greff, K., Srivastava, R.K., Koutník, J., Steunebrink, B.R., Schmidhuber, J.: LSTM: a search space odyssey. *IEEE Trans. Neural Netw. Learn. Syst.* **28**, 2222–2232 (2017)
12. Huang, P.S., He, X., Gao, J., Deng, L., Acero, A., Heck, L.: Learning deep structured semantic models for web search using clickthrough data. In: Proceedings of the 22nd ACM International Conference on Conference on Information and Knowledge Management, pp. 2333–2338. ACM (2013)
13. John, G.H., Langley, P.: Estimating continuous distributions in Bayesian classifiers. In: Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence, pp. 338–345. Morgan Kaufmann Publishers Inc. (1995)
14. Leonhardt, D., Thompson, S.A.: Trump’s lies. *N. Y. Times* **21**, 091442 (2017)
15. Liu, J., Sun, Y., Zhang, Q., Jiang, H.: Truth discovery: cotruth (2017)
16. Mathanker, S., Weckler, P., Bowser, T., Wang, N., Maness, N.: AdaBoost classifiers for pecan defect classification. *Comput. Electron. Agric.* **77**(1), 60–68 (2011)
17. Mihalcea, R., Tarau, P.: Textrank: bringing order into texts. *Assoc. Comput. Linguist.* (2004)
18. Nasukawa, T., Yi, J.: Sentiment analysis: capturing favorability using natural language processing. In: Proceedings of the 2nd International Conference on Knowledge Capture, pp. 70–77. ACM (2003)
19. Palangi, H., et al.: Deep sentence embedding using long short-term memory networks: analysis and application to information retrieval. *IEEE/ACM Trans. Audio Speech Lang. Process. (TASLP)* **24**(4), 694–707 (2016)
20. Pang, B., Lee, L., et al.: Opinion mining and sentiment analysis. *Found. Trends® Inf. Retrieval* **2**(1–2), 1–135 (2008)
21. Popat, K., Mukherjee, S., Strötgen, J., Weikum, G.: Credibility assessment of textual claims on the web. In: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, pp. 2173–2178. ACM (2016)
22. Popat, K., Mukherjee, S., Strötgen, J., Weikum, G.: Where the truth lies: explaining the credibility of emerging claims on the web and social media. In: 26th International Conference on World Wide Web. ACM (2017)
23. Quinlan, J.R.: Induction of decision trees. *Mach. Learn.* **1**(1), 81–106 (1986)
24. Riedmiller, M., Lernen, A.M.: Multi layer perceptron. Machine Learning Lab Special Lecture, University of Freiburg (2014)
25. Rose, S., Engel, D., Cramer, N., Cowley, W.: Automatic keyword extraction from individual documents. *Text Min.* 1–20 (2010)
26. Srivastava, S., Gupta, M.R., Frigvik, B.A.: Bayesian quadratic discriminant analysis. *J. Mach. Learn. Res.* **8**(Jun), 1277–1305 (2007)
27. Sundermeyer, M., Schlüter, R., Ney, H.: LSTM neural networks for language modeling. In: Thirteenth Annual Conference of the International Speech Communication Association (2012)
28. Vlachos, A., Riedel, S.: Fact checking: task definition and dataset construction. In: ACL 2014, p. 18 (2014)
29. Weikum, G.: What computers should know, shouldn’t know, and shouldn’t believe. In: Proceedings of the 26th International Conference on World Wide Web Companion, pp. 1559–1560 (2017)



SMGKM: An Efficient Incremental Algorithm for Clustering Document Collections

Adil Bagirov¹, Sattar Seifollahi^{2,3(✉)}, Massimo Piccardi²,
Ehsan Zare Borzeshi³, and Bernie Kruger⁴

¹ Federation University Australia, Ballarat, VIC, Australia

² University of Technology Sydney, Ultimo, NSW, Australia

³ Capital Markets Cooperative Research Centre, The Rocks, NSW, Australia
sseif@cmcrc.com

⁴ Transport Accident Commission (TAC), Geelong, VIC, Australia

Abstract. Given a large unlabeled document collection, the aim of this paper is to develop an accurate and efficient algorithm for solving the clustering problem over this collection. Document collections typically contain tens or hundreds of thousands of documents, with thousands or tens of thousands of features (i.e., distinct words). Most existing clustering algorithms struggle to find accurate solutions on such large data sets. The proposed algorithm overcomes this difficulty by an incremental approach, incrementing the number of clusters progressively from an initial value of one to a set value. At each iteration, the new candidate cluster is initialized using a partitioning approach which is guaranteed to minimize the objective function. Experiments have been carried out over six, diverse datasets and with different evaluation criteria, showing that the proposed algorithm has outperformed comparable state-of-the-art clustering algorithms in all cases.

Keywords: Document clustering · Incremental clustering · Spherical k -means

1 Introduction

Text document clustering has received considerable attention in the literature due to the huge amount of information generated in an electronic form in areas such as text mining and information retrieval. Given the size of such document collections, it is vital to be able to bring them into more a structured form. To this aim, *cluster analysis* can play an important role in organizing such a huge amount of documents into meaningful clusters. A cluster can be simply defined as a collection of data objects (documents here) that are ‘similar’ (under

S. Seifollahi—Currently working at Resolution Life (Australia). This work was performed while at the University of Technology Sydney.

a suitable similarity definition) to one another and dissimilar from objects in other clusters.

Most clustering methods applied to unstructured document collections start with creating a vector space known as a bag-of-words (BoW) model [24]. In this model, each document is represented by a vector with the frequencies of each word in the document. Such a representation is typically very sparse due to the large number of distinct words. The set of all documents in the vector space is usually called the document-to-term matrix.

Document clustering can be seen as a specialization of general data clustering. It was initially used for improving the precision or recall in information retrieval systems [14, 26] and as an efficient way of finding the nearest neighbors of a document [7]. Document clustering can be organized over two main stages: in the first stage, the documents are preprocessed into a usable data representation. Preprocessing may include some of the following tasks: the exclusion of words without informational value, known as stop words; the reduction of the words to their radicals, known as stemming; the uppercase/lowercase conversion, known as case-folding; and the eventual transformation into vector space. The second stage is the clustering of the vectorial representations.

Using the vector representation, various classical clustering algorithms such as the k -means algorithm and its variants, hierarchical agglomerative clustering and graph-theoretic methods have been applied in the text mining literature; for detailed reviews, see [1, 8, 11].

The similarity measure is fundamental to formalize a clustering problem. This measure, in particular, can be defined using distance(-like) functions. Clustering problems based on the squared Euclidean norm as the similarity measure are called the *minimum sum-of-squares clustering* (MSSC) problems. To date, many different algorithms have been proposed to solve this problem. Amongst them, the k -means algorithm and its variants have been amongst the most popular (see, for example, [12, 13] and references therein; and [2–4, 15, 22]).

Another similarity measure, which is widespread in text mining, is the cosine measure. The spherical k -means (SKM) algorithm [8] is the variant that uses the cosine measure. The SKM, in fact, is equivalent to a k -means algorithm using the Euclidean distance over the projection of the vector space onto the unit sphere. It has been found to work well for text clustering. The work of [5] has shown that SKM can also be derived as an EM algorithm for Maximum Likelihood Estimation of the mean direction parameters of a uniform mixture of von Mises-Fisher (or Langevin) distributions.

In the literature, there are two main categories of feature extraction methods: term frequency-based methods [8, 9, 20, 23] and semantic methods [17, 18, 28, 29]. A term frequency-based method is simply based on counting words' number, whereas a semantic method attempts to construct an ontology containing words and their relations. Term frequency-based methods tend to be simpler and more effective and, for this reason, we adopt them in this work. In particular, we leverage the term frequency-inverse document frequency (tf-idf) representation which is the product of the term frequency (tf) (the raw count of the terms appearing in the document) and the inverse document frequency (idf) which

increases the importance of terms that appear in only a few documents and conversely decreases the importance of terms appearing in many documents. The tf-idf representation has been widely utilized thanks to its computational efficiency and effectiveness [9, 20, 25].

Many document clustering algorithms such as k -means and SKM provide a means for effectively navigating, summarizing and organizing the information in the collection. In this paper, we propose an algorithm to improve the solution provided by SKM with a modest increase of computational load still within the range of a single-processor machine. Thus, our emphasis is on high accuracy with a limited increase of computational resources. The algorithm is an extension of the strategy of the incremental algorithm presented in [2], with the main difference that it projects each solution to the unit sphere to satisfy the spherical constraint. In our approach, the documents are first converted to a tf-idf representation [8, 23]. Normalizing the data vectors helps remove the bias induced by the length of a document and has generally provided superior results [8, 23, 24]. The vector space model is, then, used as an input to our two-stage algorithm. We find a better initial solution to the clustering problem in the first stage, and improve the result iteratively in the second stage by starting from the initial solution. The main contribution of our algorithm is a procedure to select better initial cluster centroids on the unit sphere in the first stage (which is different from the one in [2]), for the benefit of the ensuing MSSC optimization. The experimental results presented in Sect. 4 show that the proposed algorithm outperforms comparable one-stage algorithm spherical k -means. Unlike the spherical k -means algorithm the proposed algorithm as an incremental algorithm solves not only the clustering problem with the given number of clusters but also all intermediate clustering problems. Moreover, the proposed algorithm requires significantly less computational time than the spherical k -means algorithm in solving all clustering problems.

The rest of this paper is organized as follows. Section 2 provides a brief discussion on related works with special focus on the spherical k -means. The proposed method and related algorithms are presented in Sect. 3. Data and numerical results are reported and discussed in Sect. 4, and finally Sect. 5 contains some concluding remarks.

Throughout this paper we will use symbols m , n , and k to denote the number of documents, the number of terms (or feature), and the number of clusters, respectively. We will use symbol X to denote the set of m documents that we wish to cluster, and X^1, X^2, \dots, X^k , to denote each of the k clusters.

2 Problem Formulation

Given the vector space model, the document vectors may be represented as x^1, x^2, \dots, x^m , with each $x^i \in R^n$. Recall that m is the total number of documents and n stands for the number of unique words in the vector space model. A clustering of the document collection is its partitioning into the disjoint subsets X^1, X^2, \dots, X^k , *i.e.*

$$\bigcup_{j=1}^k X^j = \{x^1, x^2, \dots, x^m\} \quad \& X^j \cap X^l = \emptyset, \quad j \neq l.$$

In SKM, the data are projected onto the unit sphere. Dhillon et al. [8] have used the popular tf-idf scheme which reads out as (normalized) term frequency-inverse document frequency [23]. The tf-idf normalization implies that $\|x^i\| = 1$, i.e., each document vector lies on the surface of the unit sphere in R^n . The k -clustering (or k -partition) problem is formulated as the following optimization problem:

$$\begin{cases} \min & f_k(c) \\ \text{subject to} & c = (c^1, \dots, c^k) \in R^{nk}, \end{cases} \quad (1)$$

where

$$f_k(c^1, \dots, c^k) = \sum_{x \in X} \min_{j=1, \dots, k} d(c^j, x). \quad (2)$$

Here, $c^1, \dots, c^k \in R^n$ are cluster centers and the function $d: R^n \times R^n \rightarrow R_+$ is the similarity measure, R_+ is the set of nonnegative numbers.

The function f_k is called the k -th clustering objective function. The similarity measure d is defined using the cosine measure, that is for $c, x \in R^n$

$$d(c, x) = 1 - \cos(x, c) = 1 - \frac{\langle x, c \rangle}{\|x\| \|c\|},$$

where $\langle x, c \rangle$ stands for the inner product of x and c and $\|\cdot\|$ is the Euclidean norm in R^n .

Since all document vectors are normalized it is also required that for each cluster center $c^i \in R^n$ is also normalized that is: $\|c^i\| = 1, i = 1, \dots, k$. In this case one has the following similarity measure d :

$$d(c, x) = 1 - \langle x, c \rangle.$$

Then the k -clustering can be reformulated as the following constrained optimization problem:

$$\begin{cases} \min & f_k(c) \\ \text{subject to} & c = (c^1, \dots, c^k) \in R^{nk}, \\ & \|c^i\| = 1, i = 1, \dots, k. \end{cases} \quad (3)$$

The problem (3) is a nonconvex constrained optimization problem and its objective function is piecewise linear. Due to the minimum operation used in the definition of this function (see (2)), it is also nonconvex. Since the document collections usually contain hundreds of thousands of documents, objective function f_k has many local solutions. Furthermore, the typical number of words in these collections is thousands or even tens of thousands. Therefore, Problem (3) is a large-scale optimization problem. Finally, the feasible set in this problem is

nonconvex and it is a thin set in the n -dimensional space. Such problems are highly challenging not only for global optimization techniques, but also for local optimization methods. In this paper, we use the spherical k -means algorithm as our algorithm of choice.

3 The Proposed Algorithm

The proposed algorithm is based on an incremental approach. The main idea is the following: instead of working with k clusters from the start, we add the clusters one by one in successive iterations. At each iteration, we select the initial position of the added cluster by using a partitioning approach that is described in Sect. 3.1. This approach enjoys a performance guarantee that proves key for the accuracy of the overall algorithm. After the addition of the new initial cluster, a conventional k -means algorithm is called to re-optimize all the clusters (3). Then, the algorithm proceeds to the next iteration, until all clusters have been added.

3.1 Calculation of Starting Cluster Centers

The incremental algorithm proposed in this paper solves the clustering problem gradually by starting with one cluster and adding a new cluster at a time, up to the set number. Hereafter we describe the algorithm for determining the initial position of the cluster added at the k -th iteration.

Assume that the solution c^1, \dots, c^{k-1} , $k \geq 2$ to the $(k-1)$ -clustering problem is known. Denote by d_{k-1}^i the distance between x^i , $i = 1, \dots, m$ and the closest cluster center among $k - 1$ centers c^1, \dots, c^{k-1} :

$$d_{k-1}^i = \min \{d(c^1, x^i), \dots, d(c^{k-1}, x^i)\}. \tag{4}$$

We will also use the notation d_{k-1}^x for $x \in \{x^1, \dots, x^m\}$.

Consider the following two sets:

$$S_1 = \{y \in R^n : d(y, x^i) \geq d_{k-1}^i, \forall i \in \{1, \dots, m\}\},$$

$$S_2 = \{y \in R^n : \exists i \in \{1, \dots, m\} \text{ such that } d(y, x^i) < d_{k-1}^i\}.$$

The set S_1 contains all points $y \in R^n$ which do not attract any point from the set X and the set S_2 contains all points $y \in R^n$ which attract at least one point from X . It is obvious that cluster centers $c^1, \dots, c^{k-1} \in S_1$. Since the number k of clusters is less than the number of data points in the set X all data points which are not cluster centers belong to the set S_2 (because such points attract at least themselves) and therefore this set is not empty. Note that $S_1 \cap S_2 = \emptyset$ and $S_1 \cup S_2 = R^n$.

$$f_{k-1}(c^1, \dots, c^{k-1}) = \frac{1}{m} \sum_{i=1}^m d_{k-1}^i, \forall y \in S_1.$$

This means by taking any point $y \in S_1$ as a starting point for the k -th cluster center will not decrease the value of the clustering function f_k . Therefore, starting points should not be chosen from the set S_1 .

Take any $y \in S_2$. Then one can divide the set X into two subsets as follows:

$$\bar{B}_1(y) = \{x \in X : d(y, x) \geq d_{k-1}^x\},$$

$$\bar{B}_2(y) = \{x \in X : d(y, x) < d_{k-1}^x\}.$$

The set $\bar{B}_2(y)$ contains all data points $x \in X$ which are closer to the point y than to their cluster centers and the set $\bar{B}_1(y)$ contains all other data points. Since $y \in S_2$ the set $\bar{B}_2(y) \neq \emptyset$. Furthermore, $\bar{B}_1(y) \cap \bar{B}_2(y) = \emptyset$ and $X = \bar{B}_1(y) \cup \bar{B}_2(y)$.

The difference $z_k(y)$ between the value of the k -th auxiliary cluster function with the k -th cluster center y and the value $f_{k-1}(c^1, \dots, c^{k-1})$ for the $(k-1)$ -clustering problem is:

$$z_k(y) = \frac{1}{m} \sum_{x \in \bar{B}_2(y)} (d_{k-1}^x - d(y, x))$$

which can be rewritten as

$$z_k(y) = \frac{1}{m} \sum_{x \in X} \max \{0, d_{k-1}^x - d(y, x)\}. \quad (5)$$

The difference $z_k(y)$ shows the decrease of the value of the k -th cluster function f_k comparing with the value $f_{k-1}(c^1, \dots, c^{k-1})$ if the point (c^1, \dots, c^{k-1}, y) is chosen as the cluster center for the k -clustering problem.

If a data point $x \in A$ is the cluster center then this point belongs to the set S_1 , otherwise it belongs to the set S_2 . Therefore we choose a point y from the set $X \setminus S_1$. We take any $y = x \in X \setminus S_1$, compute $z_k(x)$ and introduce the following number:

$$z_{max}^1 = \max_{x \in X \setminus S_1} z_k(x). \quad (6)$$

Let $\gamma_1 \in [0, 1]$ be a given number. We compute the following subset of X :

$$\bar{X}_1 = \{x \in X \setminus S_1 : z_k(x) \geq \gamma_1 z_{max}^1\}. \quad (7)$$

If $\gamma_1 = 0$ then $\bar{X}_1 = X \setminus S_1$ and if $\gamma_1 = 1$ then the set \bar{X}_1 contains data points with the largest decrease z_{max}^1 .

For each $x \in \bar{X}_1$ we compute the set $\bar{B}_2(x)$ and its center $c(x)$. We replace the point $x \in \bar{X}_1$ by the point $c(x)$ because the latter is better representative of the set $\bar{B}_2(x)$ than the former. Denote by \bar{X}_2 the set of all such centers. For each $x \in \bar{X}_2$ we compute the number $z_k^2(x) = z_k(x)$ using (5). Finally, we compute the following number:

$$z_{max}^2 = \max_{x \in \bar{X}_2} z_k^2(x). \quad (8)$$

The number z_{max}^2 represents the largest decrease of the values $f_l(c^1, \dots, c^{l-1}, x)$ among all centers $x \in \bar{X}_2$ comparing with the value $f_{k-1}(c^1, \dots, c^{l-1})$.

Let $\gamma_2 \in [0, 1]$ be a given number. We define the following subset of \bar{X}_2 :

$$\bar{X}_3 = \{x \in \bar{X}_2 : z_l^2(x) \geq \gamma_2 z_{max}^1\}. \tag{9}$$

If $\gamma_2 = 0$ then $\bar{X}_3 = \bar{X}_2$ and if $\gamma_2 = 1$ then the set \bar{X}_3 contains only centers x with the largest decrease of the cluster function f_k .

All points from the set \bar{X}_3 are considered as starting points for solving problem (3). Therefore, their selection guarantees to maximally decrease the objective.

The algorithm for finding the initial cluster centers in solving Problem (3) can be summarized as follows:

Algorithm 1. Algorithm for finding the set of starting cluster centers.

Input: The solution (c^1, \dots, c^{k-1}) to the $(k - 1)$ -clustering problem.

Output: The set of starting cluster centers for the k -th cluster center.

Step 0. (Initialization). Select $\gamma_1, \gamma_2 \in [0, 1]$.

Step 1. Compute z_{max}^1 using (6) and the set \bar{X}_1 using (7).

Step 2. Compute z_{max}^2 using (8) and the set \bar{X}_3 using (9).

3.2 An Incremental Clustering Algorithm and Its Implementation

In this subsection we present an incremental algorithm for solving Problem (3).

Algorithm 2. An incremental clustering algorithm.

Input: The collection of documents $X = \{x^1, \dots, x^m\}$.

Output: The set of k cluster centers $\{c^1, \dots, c^k\}, k > 0$.

Step 1. (Initialization). Compute the center $c^1 \in R^n$ of the set X . Set $l := 1$.

Step 2. (Stopping criterion). Set $l := l + 1$. If $l > k$ then stop. The k -partition problem has been solved.

Step 3. (Computation a set of starting points for the next cluster center). Apply Algorithm 1 to compute the set \bar{X}_3 of starting point for the l -th cluster center.

Step 4. (Computation a set of cluster centers). For each $\bar{y} \in \bar{X}_3$ take $(c^1, \dots, c^{l-1}, \bar{y})$ as a starting point, solve Problem (3) and find a solution $(\hat{y}^1, \dots, \hat{y}^l)$. Denote by \bar{X}_4 a set of all such solutions.

Step 5. (Computation of the best solution). Compute

$$f_l^{min} = \min \{f_l(\hat{y}^1, \dots, \hat{y}^l) : (\hat{y}^1, \dots, \hat{y}^l) \in \bar{X}_4\}$$

and the collection of cluster centers $(\bar{y}^1, \dots, \bar{y}^l)$ such that

$$f_l(\bar{y}^1, \dots, \bar{y}^l) = f_l^{min}.$$

Step 6. (Solution to the l -partition problem). Set $c^j := \bar{y}^j$, $j = 1, \dots, l$ as a solution to the l -th partition problem and go to Step 2.

We call the proposed Algorithm 2 the Spherical Modified Global k -means (SMGKM). The most important and time consuming step in this algorithm is Step 4 where Problem (3) is solved starting from many initial cluster centers. For this problem, we use a conventional spherical k -means algorithm which allows us to automatically take into account the constraints of Problem (3).

4 Experimental Results

In this section we present numerical results on the evaluation of the proposed method and compare it with the Spherical k -means algorithm (SKM), described in [8]. We do not include comparison with hierarchical clustering algorithms as these algorithms have not been widely used in text mining. The reason of using the SKM for comparison is the similarity of the SKM with our proposed method in which both algorithms use spherical space and k -means as the base.

4.1 Datasets

To test and compare the proposed algorithm, we have carried out experiments with six datasets. A brief description of these datasets is given in Table 1 and more details of the datasets and preprocessing are given below.

Table 1. Dataset summary.

Datasets	m	n
1. Cora	2,240	2,319
2. Associated Press (APress)	2,246	4,994
3. WebKB	4,199	2,153
4. Reuters	12,902	1,313
5. Phone Calls (PCalls)	13,937	2,696
6. 20 Newsgroups (20Newsg)	18,774	3,103

The Cora data set [19] consists of the abstracts and references of approximately 34,000 computer science research papers; of these, we selected a subset of 2410 papers categorized into one of seven subfields of machine learning.

The Associated Press (APress) collection [10] contains Associated Press news stories from 1988 to 1990. The original data includes over 200,000 documents with 20 categories. The sample AP data set from [6], which is sampled from a subset of the TREC AP collection contains 2,246 documents.

The WebKB data set [27] consists of approximately 6000 web pages from computer science departments of various university, divided into seven categories: student, faculty, staff, course, project, department and other. In this paper, we use the four most populous entity-representing categories: student, faculty, course, and project, which all together contain 4,199 pages.

The Reuters data set [16] was originally collected by Carnegie Group, Inc. and Reuters, Ltd. in the course of developing the CONSTRUE text categorization system. It consists of 21,578 news stories appeared on the Reuters newswire in 1987. We use a subset of data containing 12,902 documents which are manually assigned to 135 categories.

The 20 Newsgroups dataset (20Newsg) [21] contains postings to Usenet newsgroups. The postings are organized by content into 20 different newsgroups with about 1000 messages from each newsgroup and are therefore well suited for text clustering. This collection consists of 18,774 non-empty documents distributed evenly across 20 newsgroups.

Finally, The Phone Calls dataset (PCalls) is from the Transport Accident Commission (TAC) which is a major accident compensation agency of the Victorian Government in Australia. It consists of a collection of 593,433 phone calls from 13,937 single TAC clients recorded by various operators over 5 years. The phone calls are made for different purposes including, but not limited to: compensation payments, recovery and return to work, different type of services, medications and treatments, pain, solicitor engagement and mental health issues.

The following preprocessing steps have been applied to all datasets before their use in the experiments: 1) removal of numbers, punctuation, symbols and “stopwords”; 2) synonyms and misspelled words have been replaced with the base and actual words (for the PCalls dataset); 3) sparse terms (95% sparsity or more) and infrequently occurring words have been removed; 4) we have also removed generic words (for the PCalls dataset) such as names and addresses based on a predefined list. The data have then been projected to a vector space by using the popular term frequency-inverse document frequency (tf-idf) scheme.

4.2 Discussion and Evaluation

Tables 2 and 3 show the best objective function value, f_{best} , and relative errors, E_1 and E_2 for the SKM and SMGKM respectively, where the relative error is defined as:

$$E_i = \frac{f_i - f_{best}}{f_{best}} \times 100$$

where f_i is the value of the clustering function obtained by i -th algorithm. In most cases, SMGKM demonstrate better performance, i.e., low values for the objective function in terms of relative errors, and in some cases the differences

are significant. When the number of clusters is small, SKM performs slightly better than SMGKM (in some cases) but the differences are not significant.

To further compare and assess the quality of the clusters generated by the algorithms, we apply two well-known cluster validity indices: the Dunn's and Davies-Bouldin validity indices.

The Dunn's validity index is defined as

$$I(D) = \max_{i=1, \dots, k} \left\{ \min_{j=1, \dots, k; j \neq i} \left\{ \frac{d(c^i, c^j)}{\max_{l=1, \dots, k} r(c^l)} \right\} \right\} \quad (10)$$

Table 2. The function value and relative errors

k	Associated press			Cora			WebKB		
	f_{best}	E_1	E_2	f_{best}	E_1	E_2	f_{best}	E_1	E_2
10	1509.66	0.00	0.15	1475.39	0.00	0.06	2607.65	0.00	0.45
12	1481.68	0.00	0.25	1455.01	0.16	0.00	2567.58	0.00	0.27
15	1456.63	0.00	0.26	1423.41	0.00	0.29	2505.60	0.00	0.09
17	1435.98	0.56	0.00	1406.54	0.00	0.21	2455.62	0.34	0.00
20	1411.64	0.66	0.00	1384.11	0.44	0.00	2408.00	0.65	0.00
25	1380.19	0.95	0.00	1351.01	0.49	0.00	2342.28	0.30	0.00
30	1355.94	0.97	0.00	1324.20	0.89	0.00	2285.21	0.25	0.00
35	1337.06	0.90	0.00	1300.97	1.28	0.00	2229.02	0.62	0.00
40	1316.64	0.95	0.00	1279.14	1.61	0.00	2183.48	0.68	0.00
45	1300.28	0.53	0.00	1262.07	1.79	0.00	2143.49	0.68	0.00
50	1286.99	1.03	0.00	1248.15	1.73	0.00	2099.22	1.01	0.00
55	1274.34	0.70	0.00	1235.44	1.95	0.00	2064.24	1.05	0.00
60	1263.40	0.85	0.00	1223.60	2.48	0.00	2035.21	1.16	0.00
65	1254.37	1.06	0.00	1213.49	1.85	0.00	2009.59	1.64	0.00
70	1245.54	0.82	0.00	1204.21	1.96	0.00	1986.34	1.58	0.00
75	1235.14	0.52	0.00	1195.76	2.13	0.00	1960.96	1.67	0.00
80	1227.43	0.78	0.00	1187.62	1.63	0.00	1939.72	1.74	0.00
85	1220.38	0.04	0.00	1180.06	1.67	0.00	1915.72	2.42	0.00
90	1213.26	0.30	0.00	1172.98	1.72	0.00	1896.69	2.43	0.00
95	1203.35	0.00	0.21	1165.74	1.88	0.00	1879.04	2.64	0.00
100	1198.26	0.00	0.13	1159.05	2.01	0.00	1862.89	2.90	0.00

where $d(c^i, c^j)$ is the distance between centers c_i and c^j . The $r(c^l)$ is the radius of the l -th cluster center and is defined as

$$r(c^l) = \max_{x \in X^l} \|c^l - x\|, \quad (11)$$

where k is the number of clusters. The Dunn’s cluster validity measure maximizes the inter-cluster distances and minimizes the intra-cluster distances. Therefore, the number of clusters that maximizes $I(D)$ can demonstrate the optimal number of the clusters.

The Davies-Bouldin validity index is a measure of within-cluster to between-cluster separation

$$I(DB) = \frac{1}{k} \sum_{i=1}^k \max_{j=1, \dots, k; j \neq i} \frac{S_k(X^i) + S_k(X^j)}{d(c^i, c^j)} \tag{12}$$

where k is the number of clusters, $S_k(X^l)$ is the average distance of all data points from the cluster X^l to their cluster center c^l and $d(c^i, c^j)$ is the distance between i -th and j -th cluster centers. Smaller values for the $I(DB)$ means that clusters are compact and far from each other. Therefore, the smaller $I(DB)$, the better clustering.

Table 3. The function value and relative errors

k	Reuters			Phone calls			20 NewsGroups		
	f_{best}	E_1	E_2	f_{best}	E_1	E_2	f_{best}	E_1	E_2
10	4586.69	0.00	0.06	9493.95	0.00	0.01	12910.54	0.01	0.00
12	4458.53	0.00	0.37	9388.49	0.00	0.16	12772.45	0.00	0.01
15	4289.97	0.00	0.02	9237.82	0.10	0.00	12590.80	0.12	0.00
17	4215.26	0.24	0.00	9144.10	0.53	0.00	12488.07	0.32	0.00
20	4119.90	0.87	0.00	9029.94	0.43	0.00	12367.28	0.15	0.00
25	3984.14	0.69	0.00	8867.60	0.20	0.00	12182.07	0.00	0.09
30	3871.80	1.17	0.00	8729.13	0.23	0.00	12042.95	0.24	0.00
35	3785.61	1.37	0.00	8614.72	0.05	0.00	11898.11	0.00	0.13
40	3719.43	1.41	0.00	8491.54	0.34	0.00	11777.05	0.00	0.27
45	3663.73	0.63	0.00	8385.29	0.23	0.00	11690.82	0.00	0.05
50	3607.86	0.84	0.00	8297.62	0.20	0.00	11559.44	0.13	0.00
55	3555.87	1.20	0.00	8214.63	0.21	0.00	11455.58	0.34	0.00
60	3512.71	1.43	0.00	8139.41	0.15	0.00	11362.46	0.30	0.00
65	3471.35	0.72	0.00	8065.04	0.00	0.06	11282.29	0.30	0.00
70	3431.43	0.95	0.00	7976.79	0.00	0.36	11200.54	0.53	0.00
75	3402.23	1.26	0.00	7931.58	0.00	0.07	11125.10	0.66	0.00
80	3374.66	0.80	0.00	7862.77	0.04	0.00	11044.67	0.60	0.00
85	3344.92	1.07	0.00	7806.99	0.18	0.00	10982.90	0.66	0.00
90	3321.59	1.16	0.00	7756.88	0.14	0.00	10926.60	0.72	0.00
95	3293.53	1.47	0.00	7707.74	0.08	0.00	10874.15	0.49	0.00
100	3271.34	1.15	0.00	7640.79	0.36	0.00	10823.80	0.63	0.00

Figures 1(a)–1(f) display the Dunn’s cluster validities for SKM and SMGKM as the number of clusters increases from 10 to 100. Here, the dot lines correspond to the SKM and solid lines to the SMGKM. Terms “dn SKM” and “dn SMGKM” stand for Dunn index values using the SKM and SMGKM, respectively. Graphs for the SMGKM are much more stable than graphs for the SKM when the number of clusters increases. The reason is likely that the SMGKM exploits an incremental scheme which adds one cluster each time, while the SKM calculates all clusters from scratch.

Figures 2(a)–2(f) show the Davies-Bouldin indices for SKM and SMGKM as the number of clusters increases. Terms “db SKM” and “db SMGKM” stand for Davies-Bouldin index values using the SKM and SMGKM, respectively. The graph for SMGKM is more stable, confirming stability in Figs. 1(a) and 1(f). These figures also demonstrate significant improvements of SMGKM over the SKM, as the graphs for SMGKM are below (much, when the number of clusters increases) those for SKM in almost all cases.

Table 4 reports the optimal number of clusters using the Dunn and Davies-Bouldin measures as well as the total CPU time spent by SKM and SMGKM. c_1^* and c_2^* stand for the optimal number of clusters using the SKM and SMGKM

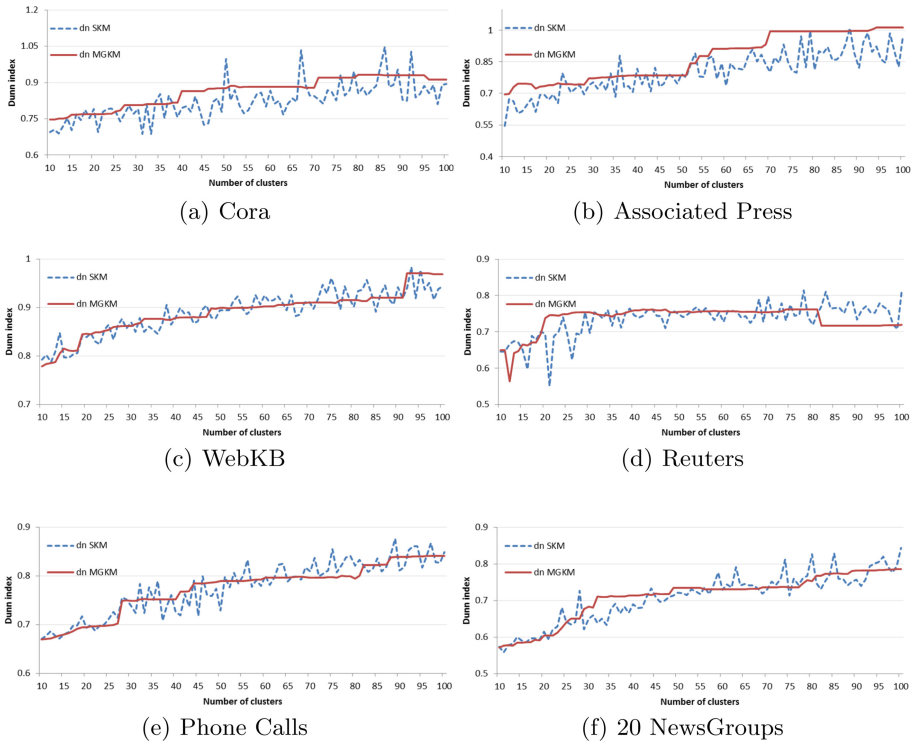
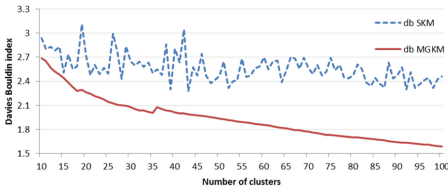


Fig. 1. Cluster validity (Dunn) index for datasets 1–6

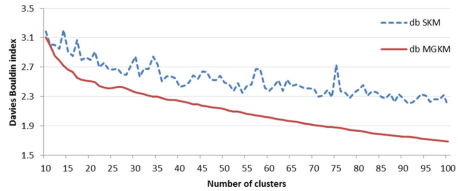
and t_1 and t_2 are the times (cumulative CPU) consumed by SKM and SMGKM, respectively. Despite using a less efficient environment for coding, the times for SMGKM have been lower than those for SKM, except on Cora.

Table 4. The optimal number of clusters and cumulative CPU time for computing up to 100 clusters. c_1^* and c_2^* stand for the number of clusters using SKM and SMGKM, respectively, and t_1 and t_2 for the time

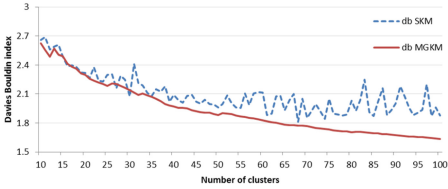
Dataset	Dunn index		DB index		Total CPU time	
	c_1	c_2	c_1	c_2	t_1	t_2
Cora	50	40	40	36	1024	1568
APress	77	70	72	74	2601	2088
WebKB	93	92	68	81	2501	1862
Reuters	29	39	10	20	5327	4745
PCalls	89	90	89	88	15861	11710
NewsG	79	81	68	77	27873	20280



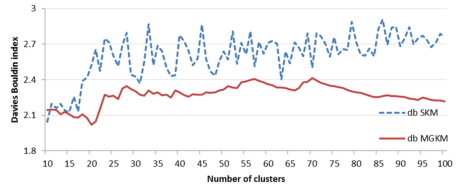
(a) Cora



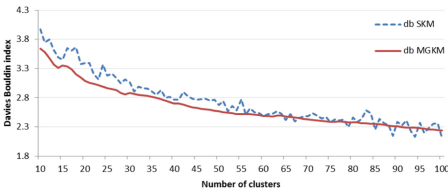
(b) Associated Press



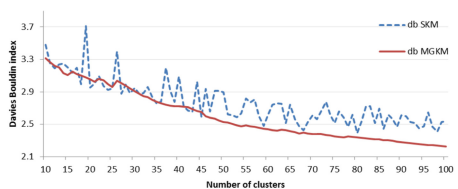
(c) WebKB



(d) Reuters



(e) Phone Calls



(f) 20 NewsGroups

Fig. 2. Cluster validity (Davies Bouldin) for datasets 1–6

5 Conclusions

In this paper, we have presented an incremental algorithm for document clustering that is capable of finding “deeper” solutions. In the algorithm, a new cluster is added in turn starting from an initial position that is guaranteed to maximally decrease the objective function value. Clustering is performed in a spherical space, meaning that each solution is projected to the unit sphere to mitigate the potential bias from more frequent words.

In the experiments, we have thoroughly compared our method with the spherical k -means algorithm (SKM) that can be regarded as state-of-the-art for document clustering. The results over six challenging datasets have shown that the proposed algorithm has consistently outperformed SKM under a number of clustering indices (objective function value, Dunn and Davies-Bouldin). This gives us ground to believe that the proposed algorithm can prove beneficial for large-scale document clustering applications.

Acknowledgement. This project was funded by the Capital Market Cooperative Research Centre in combination with the Transport Accident Commission of Victoria. Acknowledgements and thanks to industry partner David Attwood (Lead Research Partnerships). This research has received ethics approval from University of Technology Sydney (UTS HREC REF NO. ETH16-0968).

References

1. Arthur, D., Vassilvitskii, S.: k -means++: the advantages of careful seeding. In: Gabow, H. (ed.) Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms [SODA 2007], Philadelphia, pp. 1027–1035 (2007)
2. Bagirov, A.M.: Modified global k -means algorithm for minimum sum-of-squares clustering problems. *Pattern Recogn.* **41**(10), 3192–3199 (2008)
3. Bagirov, A.M., Ugon, J., Webb, D.: Fast modified global k -means algorithm for incremental cluster construction. *Pattern Recogn.* **44**(4), 866–876 (2011)
4. Bai, L., Liang, J., Sui, C., Dang, C.: Fast global k -means clustering based on local geometrical information. *Inf. Sci.* **245**, 168–180 (2013)
5. Banerjee, A., Dhillon, I.S., Ghosh, J., Sra, S.: Clustering on the unit hypersphere using Von Mises-Fisher distributions. *J. Mach. Learn. Res.* **6**, 1345–1382 (2005)
6. Blei, D., Griffiths, T., Jordan, M.I., Tenenbaum, J.: Hierarchical topic models and the nested chinese restaurant process. *Adv. Neural. Inf. Process. Syst.* **16**(106), 168–180 (2004)
7. Buckley, C., Lewit, A.F.: Optimizations of inverted vector searches. In: SIGIR 1985, pp. 97–110 (1985)
8. Dhillon, S., Fan, J., Guan, Y.: Efficient clustering of very large document collections. In: *Data Mining for Scientific and Engineering Applications*. Kluwer Academic Publishers, Oxford (2001)
9. Erra, U., Senatore, S., Minnella, F., Caggianese, G.: Approximate TF-IDF based on topic extraction from massive message stream using the GPU. *Inf. Sci.* **292**, 143–161 (2015)
10. Harman, D.: Overview of the first text retrieval conference (TREC-1). In: *Proceedings of the First Text Retrieval Conference (TREC-1)*, pp. 1–20. DIANE Publishing (1979)

11. Hartigan, J.A., Wong, M.A.: A k -means clustering algorithm. *Appl. Stat.* **28**, 100–108 (1979)
12. Jain, A.K., Murty, M.N., Flynn, P.J.: Data clustering: a review. *ACM Comput. Surv.* **31**(3), 264–323 (1999)
13. Kogan, J.: *Introduction to Clustering Large and High-dimensional Data*. Cambridge University Press, Cambridge (2007)
14. Kowalski, G.: *Information Retrieval Systems - Theory and Implementation*. Kluwer Academic Publishers, Dordrecht (1997)
15. Lai, J.Z.C., Huang, T.-J.: Fast global k -means clustering using cluster membership and inequality. *Pattern Recogn.* **43**(5), 1954–1963 (2010)
16. Lewis, D.D.: Reuters-21578 text categorization collection distribution 1.0 (1997). <http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html>
17. Liu, Y., Xiao, S., Lv, X., Shi, S.: Research on k -means text clustering algorithm based on semantic. In: *Proceedings of 10th International Conference on Computing, Control and Industrial Engineering (CCIE 2010)*, vol. 1, pp. 124–127 (2010)
18. Ma, J.: Improved k -means algorithm in text semantic clustering. *Open Cybern. Syst. J.* **8**, 530–534 (2014)
19. McCallum, A., Nigam, K., Rennie, J., Seymore, K.: Automating the construction of internet portals with machine learning. *Inf. Retrieval* **3**(2), 127–163 (2000)
20. Qimin, C., Qiao, G., Yongliang, W., Xianghua, W.: Text clustering using VSM with feature clusters. *Neural Comput. Appl.* **26**(4), 995–1003 (2015)
21. Rennie, J.: The 20 newsgroups data set (2008). <http://qwone.com/jason/20NewsGroups>, 1997
22. Ordín, B., Bagirov, A.M.: A heuristic algorithm for solving the minimum sum-of-squares clustering problems. *J. Global Optim.* **61**, 341–361 (2015)
23. Salton, S., Buckley, C.: Term-weighting approaches in automatic text retrieval. *Inf. Process. Manag.* **24**(5), 513–523 (1988)
24. Salton, G., McGill, M.J.: *Introduction to Modern Retrieval*. McGraw-Hill Book Company, New York (1983)
25. Seifollahi, S., Bagirov, A., Layton, R., Gondal, I.: Optimization based clustering algorithms for authorship analysis of phishing emails. *Neural Process. Lett.* 1–15 (2017)
26. Van Rijsbergen, C.J.: *Information Retrieval*, 2nd edition. Butterworth, London (1989)
27. WebKB: Available electronically at <http://www.cs.cmu.edu/~WebKB>
28. Yi, J., Zhang, Y., Zhao, X., Wan, J.: A novel text clustering approach using deep-learning vocabulary network. *Math. Probl. Eng.* **1**, 1–13 (2017)
29. Zhang, W., Yoshida, T., Tang, X.: A comparative study of TF-IDF, LSI and multi-words for text classification. *Expert Syst. Appl.* **38**, 2758–2765 (2011)



From Emoji Usage to Categorical Emoji Prediction

Gaël Guibon^{1,2}(✉), Magalie Ochs¹, and Patrice Bellot¹

¹ LIS-CNRS UMR 7020, Aix-Marseille Université, Marseille, France
{gael.guibon,magalie.ochs,patrice.bellot}@lis-lab.fr

² Caléa Solutions, Marseille, France

Abstract. Emoji usage drastically increased recently, they are becoming some of the most common ways to convey emotions and sentiments in social messaging applications. Several research works automatically recommend emojis, so users do not have to go through a library of thousands of emojis. In order to improve emoji recommendation, we present and distribute two useful resources: an emoji embedding model from real usage, and emoji clustering based on these embeddings to automatically identify groups of emojis. Assuming that emojis are part of written natural language and can be considered as words, we only used unsupervised learning methods to extract patterns and knowledge from real emoji usage in tweets. Thereby, emotion categories of face emojis were obtained directly from text in a fully reproducible way. These resources and methodology have multiple usages; for example, they could be used to improve our understanding of emojis or enhance emoji recommendation.







Keywords: Emoji · Recommendation · Word embeddings · Resource · Clustering

1 Introduction

Research on emojis are growing steadily since a couple of years. Nowadays users of instant messaging applications have to scroll through huge libraries of emojis to select one: it would be useful to help the users by automatic recommendation. However, several emojis can be used to convey the same emotion, for instance 😊 and 😄, and it is not clear whether or not emojis can be considered as actual words, groups of metadata, or extra linguistic information for Natural Language Processing (NLP) approaches. Most of the recommendation systems tackle emoji as metadata. In this paper, we propose an approach considering emoji as words without any assumptions on their meanings.

Nowadays, few emoji recommendation systems have been proposed, and only very recently. Moreover, actual emoji recommendation systems consider each emoji as a single label to recommend. Those systems obtained perfectible results: 65% accuracy [20] and 65% f1-score [2]. These systems focused only on a very


limited number of emojis. Considering these works, instead of recommending emojis we propose to automatically recommend groups of emojis. Our work is based on the following hypotheses:

1. Emojis can be quite similar (   ) and it is not clear whether we should recommend an emoji instead of another, thus we should predict categories of emojis instead of only considering specific ones
2. Emoji categories should be inferred from their real usage in order to adapt to changes and cultural differences in order to obtain a good recommendation
3. Emotion-related emojis cannot be recommended only by matching keywords, as current smartphone systems implement for   for instance. Emotion-related emojis can come from the whole feeling of the text. This is why we focus on them at first.

Considering these hypotheses, our purpose in this paper is to automatically regroup emojis based on their usage, focusing on the common subset of 63 face emojis. We propose an automatic emotion categorization of face emojis not by using metadata, but by using the real context usage of emojis in order not to assume predefined informations. The methodology used to obtain these resources can be applied on any emoji type, even though we applied it to face emojis: it consists in analyzing the similarity of context usage between emojis by using two unsupervised approaches: first, word embeddings of only tweets containing emojis (Sect. 3), then using these embeddings to apply a clustering algorithm (Sect. 4).

The paper is organized as follow: we summarize the related work on emojis and emoji embeddings (Sect. 2) before presenting our first resource: emoji embeddings (Sect. 3). Then, we present the second resource, a fine-grained clustering of face emojis (Sect. 4) before validating it with Ekman's theory on emotion-face expressions (Sect. 4.2).

2 Related Work

Emoticons (:-) , :P) and emojis () are two different ways to represent facial cues. While the former are characters, the latter are pictures, and as such they can also represent different concepts and ideas, not only facial cues and expressions. Moreover, emojis tend to replace emoticons in social conversations [16]. The first 176 emojis were released in 1999 by the japanese telecom NTT DOCOMO¹, right now there are 2623 emojis according to the official Unicode list². Their success is due to the support of emojis by the first iPhone from Apple, then other brand such as Google or Samsung started supporting them.

¹ <https://www.nttdocomo.co.jp/>.

² <http://unicode.org/emoji/charts/full-emoji-list.html>.

At first, some research work in socio-linguistics focused on the diverse understanding and usage of emojis, and the role they have in a textual conversation. According to these, emojis are used to improve the understanding of the message in 70% of cases [9]. Also, it has been demonstrated that emojis can serve a number of roles in conversation [10] which are not necessarily related to the expression of emotions, such as maintaining a conversational connection, or engaging in playful interaction. Some of these roles can be linked to the Jakobson’s functions of language [8]:

- Phatic function: “👉”
- Referential function: “Just bought it 🚲”
- Emotive function: “Seriously?! 🤔”
- Conative function: “👉” for “call me”

Thus, it is necessary to have a good emoji recommendation system in social messaging application in order to enhance the conversation’s quality.

Few research works focused on emoji recommendation, the main approach is based on neural networks to predict predict emojis. However, the performance of these models remains perfectible. Xie *et al.* [20] achieved 65% of accuracy for the 3 mostly used emojis in conversations from Weibo³, the chinese Twitter, using Hierarchical LSTM [11]. Barbieri *et al.* [2] predicted the 20 most used emojis in 40 millions tweets using Long Short-term Memory Networks [7] and evaluated their prediction on the 5 most frequent emojis obtaining an average f1-score of 65%.

They are still only a few available resources for research on emoji, whether it is emoji embedding models or other kind of resources. Considering embedding models, there have been only a few work on emoji embeddings and all of them have been done recently. On one hand some work do not embed emojis directly. Emoji has been considered to be a group of meta-informations or words upon which the embedding will be based on. These meta-informations can be Unicode descriptions of emojis [4] or their possible meanings and senses associated to their description [1, 19].

On the other hand, emoji have been embedded directly in context with all types of emojis from millions of tweets. In this paper, we used similar approach. However, in existing works, arbitrary clusters have been defined [3] or hierarchial ones mixing all types of emojis either to detect sarcasm [6] or to find the best keyboard mapping [17]. The resources proposed in this paper are created totally automatically without predefined clusters and focusing on a particular type of emoji: the face emojis. The face emojis correspond in the Unicode list⁴ to the ”face positive”, ”face neutral”, and ”face negative” categories, which reflect faces with emotional expressions⁵.

Our objective is to automatically obtain emoji clusters from real usage in order to further recommend them. We can compare our work with the one from

³ <http://www.weibo.com/>.

⁴ <http://unicode.org/emoji/charts/full-emoji-list.html>.

⁵ As a first step, we did not considered animal faces.

Pohl *et al.* [17] which tackled hierarchical relations between emojis of any kind. In the contrary to our work, they did not tried to obtain clusters inside emojis specific types to enhance emoji recommendation systems, which is our main purpose by applying our methodology and obtaining these new resources for further emoji prediction models.

3 Emoji Embeddings

The most used emojis are those representing emotions or sentiments, according to their usage in social media such as Twitter⁶: 😊❤️😄❤️😭.

Considering this fact, we want to verify if face emoji usages implicitly follow existing face expression categorizations by observing the usage of the 63 face emojis, excluding cat 🐱, demon 😈, alien 👽 emojis and so on 🧑. We make an emoji embedding in order to obtain a more fine-grained categorization for these emojis. We exclude the very recent emojis which are not in our dataset such as the exploding face 💣.

3.1 Dataset

Our dataset is made of 695 031 tweets emitted from the North American continent (United States and Canada), all of them containing at least one of the 800 emojis from our list, and all collected using the Twitter streaming API⁷. There is no topic filtering so all kind of topics are included. The dataset is composed of tweets in english using a language detection process made with the occurrence ratio of NLTK stopwords list⁸. Table 1 shows quantitative information on the dataset.

In the dataset, we consider emojis as words. Thus, we keep them as tokens. To ensure they are used as tokens we tokenized the text and separate each emoji from other word. Then we applied lemmatization to the words using NLTK. Hence, we obtained a corpus of tweets ready for applying unsupervised algorithms.

Table 1. Dataset of tweets containing emojis

Tweets	695 031
Emojis	901 669
Average tweet length	10.81 words
Distinct Emojis	844
Emoji/tweets	1.30

⁶ <http://www.emojitracker.com/>.

⁷ <https://dev.twitter.com/streaming/overview>.

⁸ <http://www.nltk.org/>.

3.2 Embeddings

To embed tweets with their emojis we used two approaches using Word2Vec [14, 18] in its gensim implementation⁹. We used a Continuous Bag-Of-Words (CBOW) embedding to predict target (emoji) from context words (tweet) with hierarchical softmax [13], and another embedding using Skip-grams to predict a context using an emoji. For comparison, we also used the skip-grams embedding model from Pohl *et al.* [17]. The resulting vector spaces are made of 300 dimensions from words with a minimum of 5 occurrences. These different models are then used to compare their impact on the clustering (Sect. 4).

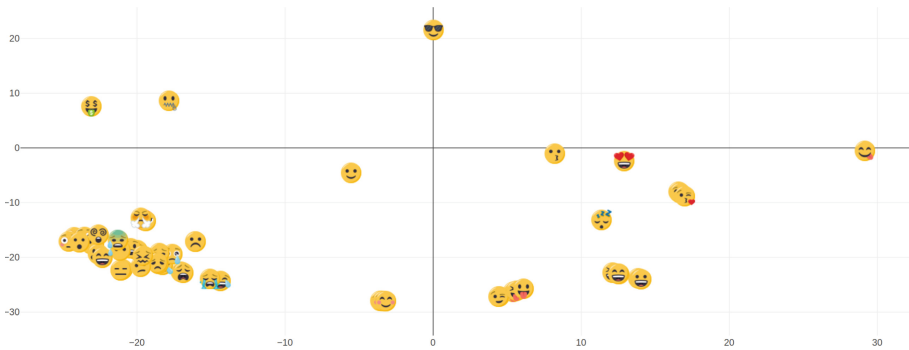


Fig. 1. Embeddings of 63 face emojis.

To display the dimensional space we selected the 63 emojis we are focusing on. To display a 300-dimensional space, we used the T-distributed Stochastic Neighbor Embedding (TSNE)¹⁰ [12] in its Scikit-Learn implementation¹¹ to reduce this high dimensional space to a 2 dimensions space. The distribution of the emojis in the resulting 2 dimensions space is visible in Fig. 1.

The complete 2 dimension visualization of the embedding space are available as supplementary materials and show different groups. However, the visualization can be quite different depending on the TSNE parameters, making the visualization not reliable to induce emoji groups. For the TSNE parameters we used a learning rate of 100, a perplexity of 30, and an early exaggeration of 2.0. Other parameters are the default ones from its Scikit-learn implementation (See footnote 13). Another approach would consist in defining an arbitrary threshold for the cosine distance to create clusters. To avoid arbitrary decisions, or group section based on visual proximity, we decided to apply clustering on the produced emoji embeddings (Sect. 4), without assuming a number of clusters.

⁹ <https://radimrehurek.com/gensim/models/word2vec.html>.

¹⁰ <https://lvdmaaten.github.io/tsne/>.

¹¹ <http://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html>.

4 From Embeddings to Emoji Categories

4.1 Clustering

In the previous section we chose an emoji vector space of dimension 300. However, it does not give clusters. As we wanted to automatically cluster face emojis from real usage data, we need to avoid any human interaction in the process. Hence, instead of directly using the cosine similarity we applied two clustering algorithms to automatically identify emoji clusters.

Unlike existing emoji clusterings in which the clusters were constructed based on the text and the emojis [3, 17], we applied clustering only on emoji vectors, even if they have been embedded using raw text. Plus, as we consider a sub type of emojis, this allows us to automatically retrieve good clusters directly from source, without mixing emoji types.

Clustering Algorithms Applied to Emoji Embeddings. We first applied the k-means algorithm to compute cluster centroids from the dataset considering n clusters = n labels. So we did not assume a predefined number of clusters, each element could be contained into only one cluster. The k-means parameters were 63 possible clusters (for 63 emojis) with a maximum of 500 iterations and a number of 1000 initialisations. At the end, we removed the empty clusters and we finally obtained a reduced number of 18 clusters.

In order to compare the results with another algorithm, we used spectral clustering algorithm [15] considering 63 possible clusters. The hyper parameters were as follow: no eigenvalue decomposition strategy, a gaussian kernel, a gamma of 0.7, and discretization to assign label in order to differ from k-means. At the end, spectral clustering also gave us a total of 18 clusters.

K-means and spectral clustering results were close but we decided to keep the 18 clusters from spectral clustering because it avoided splitting intuitive emoji clusters such as kisses emojis 🍷 🍷 🍷 🍷. The different clusters are provided in the supplementary materials of the paper¹².

Using a skip-gram emoji embedding model we obtained 11 coarse clusters¹³. The resulting clusters are not satisfying since they mixed several emojis representing different emotions: anger, love, and tiring faces being in the same cluster for instance. With CBOW embeddings we obtained 18 emoji clusters¹⁴, being more specific and fine-grained. Some of these clusters are visible in Table 2.

Comparison with Existing Related Work. In order to compare the resulting clustering depending on the emoji embeddings, we also used Pohl *et al.* [17] embedding model, learnt using skip-gram, to extract face emoji clusters the same way we did using our embeddings. We used the same methodology, the desired number of clusters being equal to the number of elements, *i.e.* 63. The resulting

¹² See the html files in the “visualization” folder.

¹³ “clusters_native63_skipgram.html” in supplementary materials of the paper.

¹⁴ “clusters_native63_cbow.html” in supplementary materials of the paper.

Table 2. Emoji clusters using Spectral Clustering on CBOW embeddings. Labelled using Ekman’s categories of facial emotion expressions.

None (sleep) 😴	None (closed mouth) 😬
Contentment (Kisses) 😘😗😙😚	Sadness 😞😓😭😮😱😲😳😴😵😶
Contentment 😄😃	Shame 😞😓
Excitement 😄	None (not clear) 😞😓
Anger 😡😠😤	Amusement 😄😃😆😅😂
None (Greed) 😏	Sensory Pleasure 😄😃😆😅
Fear / Surprise 😱😨😧😮😲😳😴	Contentment / Amusement 😄😃😆
Satisfaction / Pride in Achievement 😄😃😆	None (Not happy) 😞😓😭😮
Contempt 😏😒😓😔	Excitement (Love) 😄

clusters are not satisfying and can be seen in Table 3. There were merely 6 coarse clusters being merely global separation between joy, anger, surprise and sadness. Considering the coarse clusters obtained using both skip-gram models and the fine-grained clusters obtained using a CBOW embedding model, we came to the conclusion that CBOW embeddings are better to obtain more specific clusters of emojis. Hence, unlike existing emoji embeddings all using skip-gram models, this approach is better to retrieve latent information about real emoji usage in short messages.

Table 3. Clusters obtained using Pohl *et al.* embeddings

😄😃😆😅😂😇😈😉😊😋😌😍😎😏😐😑😒😓😔😕😖😗😘😙😚😛😜😝😞😟😠😡😢😣😤😥😦😧😨😩😪😫😬😭😮😯😰😱😲😳😴😵😶😷😸😹😺😻😼😽😾😿
😞
😡😠😤😥😦
😱😨😧😮😲😳😴
😄😃😆😅
😞😓😭😮

Because we consider the face emojis as our study material in this paper, we tried to obtain labels for these clusters considering existing theory (Sect. 4.2).

4.2 Cluster Validation Through Ekman’s Theory

As the clusters represent face expressions, we can consider we did an face expression categorization of text messages. To verify if this categorization represents known emotions and evaluating their quality, we decided to labellize these clusters by Ekman’s 16 basic face expressions of emotions [5] in order to compare and find the mistakes. For this purpose, we took each cluster automatically obtained and linked it to one of Ekman’s categories by identifying the similarities on the emojis faces and the Ekman’s facial expressions. For instance, the sadness category is represented by 🙄😞😓😭😔😩😫😬😪😧😨😱😲😳😴.

The labelled clusters are described in Table 2. Some categories are splitted by intensity, such as the joy wich have two clusters automatically extracted: one for the mild contentment 😊😄. 😁😂😃. Also, some Ekman’s emotion categories can overlap in those clusters, such as fear/surprise.

Moreover, only the two unrealistic emojis and the sleeping emoji were alone in their clusters: 🤪, 🤩 and 😴.

Note that the label attributed to the clusters may be discussed. However, the objective is not to find the precise label for each cluster but to identify different clusters of emoji. A comparison with the work of Ekman enabled us to relate emotional face expressions of humans to virtual facial expressions of emojis. This comparison shows that emojis are somewhat used the same way the face expressions of emotions are. However, some specific categories are inherent of emojis, such as 🤪 and 🤩.

5 Conclusion and Perspectives

In conclusion, in this paper we presented two resources: an novel emoji embedding in real context in the scope of face emojis, and a set of face emoji automatic clusters from real usage only. Both can be used to improve emoji recommendation systems: instead of recommending specific emojis, groups of emojis (corresponding to clusters) will be used as elements to recommend. Recommending clusters of face emojis will positively impact the recommendation quality, as face emojis are some of the most used emojis. Moreover, the methodology we used to obtain these resources can be reproduced on different types of emojis to identify inherent categories from their usage. For instance, vehicule emoji unsupervised categorization could be done.

The methodology and resources can be used to recommend the emotion categories to express by an embodied conversational agent or in general dialog system, such as trending chatbots. With this work we want to change how to tackle emoji prediction by trying to generalize more, not only by parameter tuning, but also by changing the scope of the recommendation. Of course, because we focused on automatically retrieving emotion clusters of face emojis, theses resources could be helpful out of the scope of recommendation. For instance, embodied conversational agent could use them to determine which face expression is relevant to which emotion, and how to reproduce them without having to

necessarily regroup them from theories. Making the conversational agent more adaptative.

Finally, these resources with their visualisations and the python code used to produce them are available as supplementary materials. We plan to make them available to everyone in ORTOLANG¹⁵, a platform that is part of CLARIN infrastructure¹⁶. The code and links to the models are available in the following repository: <https://github.com/gguibon/FaceEmojis>.

References

1. Ai, W., Lu, X., Liu, X., Wang, N., Huang, G., Mei, Q.: Untangling emoji popularity through semantic embeddings. In: ICWSM, pp. 2–11 (2017)
2. Barbieri, F., Ballesteros, M., Saggion, H.: Are emojis predictable? arXiv preprint [arXiv:1702.07285](https://arxiv.org/abs/1702.07285) (2017)
3. Barbieri, F., Ronzano, F., Saggion, H.: What does this emoji mean? a vector space skip-gram model for twitter emojis. In: Language Resources and Evaluation conference, LREC. Portoroz, Slovenia (2016)
4. Eisner, B., Rocktäschel, T., Augenstein, I., Bošnjak, M., Riedel, S.: emoji2vec: learning emoji representations from their description. arXiv preprint [arXiv:1609.08359](https://arxiv.org/abs/1609.08359) (2016)
5. Ekman, P.: Basic emotions in t. dalglish and t. power (eds.) the handbook of cognition and emotion, pp. 45–60 (1999)
6. Felbo, B., Mislove, A., Søgaard, A., Rahwan, I., Lehmann, S.: Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. arXiv preprint [arXiv:1708.00524](https://arxiv.org/abs/1708.00524) (2017)
7. Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural Comput.* **9**(8), 1735–1780 (1997)
8. Jakobson, R.: Closing statements: Linguistics and poetics, style in language. TA Sebeok, New York (1960)
9. Kelly, C.: Do you know what i mean >:(a linguistic study of the understanding of emoticons and emojis in text messages (2015)
10. Kelly, R., Watts, L.: Characterizing the inventive appropriation of emoji as relationally meaningful in mediated close personal relationships. In: Experiences of Technology Appropriation: Unanticipated Users, Usage, Circumstances, and Design (2015)
11. Li, J., Luong, M.T., Jurafsky, D.: A hierarchical neural autoencoder for paragraphs and documents. arXiv preprint [arXiv:1506.01057](https://arxiv.org/abs/1506.01057) (2015)
12. Maaten, L.V.D., Hinton, G.: Visualizing data using t-sne. *J. Mach. Learn. Res.* **9**(Nov), 2579–2605 (2008)
13. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint [arXiv:1301.3781](https://arxiv.org/abs/1301.3781) (2013)
14. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013)
15. Ng, A.Y., Jordan, M.I., Weiss, Y.: On spectral clustering: analysis and an algorithm. In: Advances in Neural Information Processing Systems, pp. 849–856 (2002)

¹⁵ <https://www.ortolang.fr/>.

¹⁶ <https://www.clarin.eu/>.

16. Pavalanathan, U., Eisenstein, J.: Emoticons vs. emojis on twitter: a causal inference approach. arXiv preprint [arXiv:1510.08480](https://arxiv.org/abs/1510.08480) (2015)
17. Pohl, H., Domin, C., Rohs, M.: Beyond just text: semantic emoji similarity modeling to support expressive communication. *ACM Trans. Comput.-Human Interact. (TOCHI)* **24**(1), 6 (2017)
18. Rehurek, R., Sojka, P.: Software framework for topic modelling with large corpora. In: *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Citeseer (2010)
19. Wijeratne, S., Balasuriya, L., Sheth, A.P., Doran, D.: Emojinet: an open service and api for emoji sense discovery. In: *ICWSM*, pp. 437–447 (2017)
20. Xie, R., Liu, Z., Yan, R., Sun, M.: Neural emoji recommendation in dialogue systems. arXiv preprint [arXiv:1612.04609](https://arxiv.org/abs/1612.04609) (2016)

Text Generation



Towards Social Context Summarization with Convolutional Neural Networks

Minh-Tien Nguyen^{1,2(✉)}, Duc-Vu Tran¹, Viet-Anh Phan¹,
and Le-Minh Nguyen¹

¹ Japan Advanced Institute of Science and Technology (JAIST), 1 -1 Asahidai,
Nomi, Ishikawa 923 -1292, Japan

{[tienm.vu.tran](mailto:tienm.vu.tran@jaist.ac.jp), [anhphanviet](mailto:anhphanviet@jaist.ac.jp), [nguyenml](mailto:nguyenml@jaist.ac.jp)}@jaist.ac.jp

² Hung Yen University of Technology and Education, Hung Yen, Vietnam

Abstract. This paper studies Web document summarization by exploiting social information. The motivation comes from the fact that social context of a Web document provides additional information to enrich the content of sentences. To take advantage of such information, we design a model based on Convolutional Neural Networks. Unlike traditional ones, our model enhances representation of sentences by mutually combining internal and social information. The model learns to rank sentences and user posts and then extracts top ranked ones as a summary. Experimental results on three datasets in two languages confirm the efficiency of our model in summarizing single documents.

Keywords: Summarization · Convolutional Neural Networks

1 Introduction

In the context of social media, there exists a relationship between a document and its relevant information. For example, after reading the *Asiana Airlines Flight 214 crash* event, readers can write their comments on the event by using the Web interface or post tweets on their Twitter timeline. In the meantime, the content of the main document can be found in relevant articles published by different news providers. The combination of Web documents and their related information defines a summarization task which takes advantage of social information to extract high-quality summaries [3, 17, 21, 23].

Social Context. The social context of a Web document d is $C_d = \langle S_d, R_d, UP_d \rangle$, where S_d is a set of sentences in d , R_d is relevant articles of d , and UP_d is a set of user posts, e.g. tweets or comments of d [23].

The Task. Given a document d and its context C_d , the task is to extract m important sentences from S_d and m representative user posts from UP_d [17, 21, 23]. The intuition is that user posts can enrich extracted sentences in providing new information which is not usually available in the main document.

This paper introduces a model based on Convolutional Neural Networks (CNN) [6] to build a summarizer, which exploits social context of Web documents

to extract high-quality summaries. The motivation comes from the fact that social context provides additional valuable information for enriching the meaning of primary documents. To take advantage of such information, we employ CNN to learn hidden features of sentences and user posts. More precisely, our model captures n -grams and combines them to enhance hidden representation. To integrate the support of external information, we incorporate surface features into the hidden representation to create final vectors. Finally, our model learns to rank sentences and user posts and extracts top m ranked ones as a summary. This paper makes the following main contributions:

- It studies the task of summarizing Web documents by exploiting their social information with the use of CNN. To the best of our knowledge, no existing methods address this task by using CNN.
- It provides a way to enhance the representation of sentences and user posts, which benefits the learning process. The enhancement is in two levels: enriching hidden representation learned by the model and integrating social information in the form of features.
- It investigates the influence of features, which reveals the contribution of each information channel in our model and releases two improved datasets including related articles of main documents.¹

We apply our model to the task of sentence and highlight extraction on three datasets, in English and Vietnamese. Our model achieves competitive ROUGE-scores compared to advanced methods in summarizing Web documents.

2 Related Work

Web document summarization has recently been enriched by using social information [4, 9, 20, 23]. Researchers integrate social information by using supervised methods to build dual wing factor graphs [23], or presenting features for modeling cross relationships between sentences and tweets [21]. By contrast, several unsupervised models have been also presented [3, 7, 22] such as building a cross-collection topic-aspect for a co-ranking method to extract sentences and tweets [3], creating heterogeneous graphs of random walks to summarize single documents [22], using integer linear programming [7], or matrix co-factorization [12].

The system of Nguyen et al. [15, 17] is perhaps the most relevant work to our study. It integrates sentences, relevant social messages, and related articles in a unified model, which extracts features from three information channels. The authors use CRF and Ranking SVM to train summarizers to output scores used to select top-ranked sentences and user posts. In this work, we extend the work of [15, 17] by adapting a different learning algorithm based on CNN. We also encode social context into our model by integrating sophisticated features adopted from [15, 17] to enrich the representation of sentences and user posts.

¹ We provide main codes, features, and data at <https://goo.gl/rC6C8n>.

3 Summarization with CNN and Social Context

This section shows our proposal to address the task in three steps: data preparation, sentence ranking and selection, and the settings of experiments.

3.1 Data Preparation

Since DUC datasets lack social information, we used three other ones for social context summarization. *SoLSCSum* [14] is an English dataset collected from Yahoo News, containing sentences and comments, which were manually annotated. To validate our model in a non-English language, we used a Vietnamese dataset created for social context summarization named *VSoLSCSum* [13]. It was collected from several Vietnamese Web pages.²

Table 1. Statistical observation on the two datasets.

Dataset	#doc	#sentences	#comments/tweets	#refs	#relevant docs
SoLSCSum	157	3,462	25,633	5,858	1,570
VSoLSCSum	141	3,760	6,926	2,448	1,410
USAToday-CNN	121	6,413	78,419	455	1,210

We also used *USAToday-CNN* [18] derived from [21] for news highlight extraction. The dataset contains 121 events collected from USAToday and CNN.

We followed [15] in exploiting relevant Web documents as third-party sources to enrich feature extraction. To do that, we used the provided dataset of SoLSCSum from [15]. It includes primary documents, relevant user posts, and related articles. For VSoLSCSum and USAToday-CNN, we used guideline from [15] to retrieve top 10 relevant articles from Google by searching the title of the main document (Table 1). Finally, each dataset contains three parts: main documents, relevant user posts (comments/tweets), and related articles.

3.2 Sentence Ranking

Our model receives documents and their social context for ranking. To do that, we adapted CNN for our summarization task because it has shown the ability to effectively learn n -gram sequences, which help to generate rich textual representation and to tackle sentences with variable lengths naturally. For ranking, our model combines two types of features: (i) latent textual features generated from CNN with customized pooling operation and (ii) surface features extracted from social context. The combination of the two types allows our model to learn the representation of sentences and user posts effectively. We next describe our model in two steps: convolution and polling, and social context integration.

² We acknowledge [13, 14] for sharing data partition of SoLSCSum and VSoLSCSum.

Convolution and Pooling. Our model adapts CNN with customized pooling operation for generating latent features, representing textual information from both sentences and user posts.

Convolution. Let h is the kernel size (window size) and $M \in \mathbb{R}^{N \times k}$ is an embedding matrix where N is the max sentence length and k is the dimension of word embedding, the convolution operation applies the kernel size h to each position i of M to produce non-linear transformation of the input [6].

$$c_i^h = f(\mathbf{W}^h \cdot v_i + b) \quad (1)$$

where $f()$ is a non-linear function, e.g. $\tanh()$, \mathbf{W}^h is the kernel weights, b is the bias term, and v_i is an input corresponding to a sub-region of size h . The feature map of a kernel with size h is $\mathbf{C}^h = [c_1^h, \dots, c_{N-h+1}^h]$. To obtain rich representation, we adopted multi-window-size filters from Kim. (2014) [5] in order to formulate n -grams of an input sentence.

Pooling. The pooling is the second layer of a CNN-based model, which receives the output of the convolution. It includes down-sampling operation to modify the output of the convolution for retaining most important features from a feature map. For our purpose, we adapted the pooling operation in two phases: (i) max-over-time pooling and (ii) cross-window pooling. The first pooling outputs the most essential features \hat{c}^h from the feature map \mathbf{C}^h .

$$\hat{c}^h = \max\{\mathbf{C}^h\} \quad (2)$$

The second pooling phase applies a max-pooling function over representation from different window size filters. Our intuition is that the model can, across window sizes, obtain richer hidden representation which benefits the learning process. We denote this operation as cross-window max-pooling, which extracts rich and representative information across all window sizes. Let $\hat{\mathbf{C}} = \{\hat{c}^h\}$ be a set of all different important features of window sizes h , and $\hat{\mathbf{C}}_1^k, \dots, \hat{\mathbf{C}}_N^k$ be all the size- k subsets of $\hat{\mathbf{C}}$ where $|\hat{\mathbf{C}}_i^k| = k$. The output of the second pooling is:

$$x_p = [\max\{\hat{\mathbf{C}}\}; \max\{\hat{\mathbf{C}}_1^k\}; \dots; \max\{\hat{\mathbf{C}}_N^k\}] \quad (3)$$

The vector x_p is the concatenation of latent features extracted by concatenating all feature maps $\hat{\mathbf{C}}$ and a subset of feature maps. Our pooling operation shares the idea with [1] which also applies pooling on different window sizes. However, their pooling operation only selects one value from all window sizes leading to least information kept. By contrast, our pooling operates not only on the all but also on the subsets of the windows sizes. This allows our model to enhance the hidden representation of sentences.

Social Context Incorporation. As discussed, there exists relationships between Web documents and their related information, e.g. relevant social messages and articles. We, therefore, take advantage of such related information to

enrich the hidden representation learned by CNN. To do that, we adopted features from [15]. Given a sentence in the main document, we extracted three types of features from three channels: local features, relevant user-posts features, and third-party features (related articles). Local features individually measure the importance of a sentence in the main document (or a comment/tweet), without considering relevant user posts or third-party sources. User-post features present relationships between sentences in a primary document and its user posts. Third-party features estimate the importance of a sentence (or a user post) by using relevant documents. Table 2 shows our features.

Suppose \mathbf{x}_e is the concatenation of three vectors \mathbf{x}_l , \mathbf{x}_s , and \mathbf{x}_t denoted for the three feature groups correspondingly, the final representation of a sentence (or a user post) is the concatenation of \mathbf{x}_p and \mathbf{x}_e .

Table 2. Local (LF), social (SF), and third-party features (TPF).

Group	Feature	Description
LF	Position	The sentence position of x_i in a document (pos = 1, 2, or 3)
	Length	The number of terms appearing in s_i after removing stop words
	Title-words	#common words in s_i and the title after removing stop words
	#Stopwords	Number of stopwords in a sentence s_i
	HIT-score	The HIT score of a sentence s_i
	LSA-score	The LSA score of a sentence s_i
	Cosine	Cosine similarity with N next and previous sentences ($N=1,2,3$)
	Them-words	Count #frequent words calculated by TF appearing in x_i
	In-words	Whether x_i contains indicator words appearing in a dictionary
Up-words	Whether x_i contains upper case words e.g., proper names	
UPF	Max-cosine	Maximal Cosine of a sentence s_i with social information
	Max-dist	Maximum distance of a sentence s_i with social information
	Max-lexical	Maximal lexical score of s_i with social information
	Max-W2V	Maximal W2V score of a sentence s_i with social information
TPF	Voting	#sentences in relevant documents having Cosine \geq a threshold regarding to s_i
	Cluster-dist	The Euclidean distance from s_i to relevant documents
	stp-TF	The score of s_i in relevant documents calculated by TF-IDF
	aFrqScore	The average probability of frequent words in relevant documents regarding to s_i
	frqScore	The probability of frequent words in supporting documents regarding to s_i
rFrqScore	The relative probability of frequent words in supporting documents regarding to s_i	

$$\mathbf{x}_e = [\mathbf{x}_l; \mathbf{x}_s; \mathbf{x}_t] \quad (4)$$

$$\Theta = [\mathbf{x}_p; \mathbf{x}_e] \quad (5)$$

By integrating indicators extracted from three channels, our model not only enriches vector representation but also incorporates social context into the summarization process. The final vector representation Θ of a sentence s_i is fed to two MLP layers for regression. Also, note that we present sentences and user posts in a mutual support fashion. When modeling a user post, sentences and related articles are used as supporting information with the same feature set.

3.3 Sentence Selection

We separately trained two models, for sentences and user posts. We used scores generated from the models for ranking. Summaries are extracted by selecting top m ranked sentences and user posts.

3.4 Settings and Evaluation Metrics

Settings. We used 10-fold cross-validation for SoLSCSum with $m = 6$ as the suggestion of [14]; 5-fold cross-validation for VSoLSCSum and USAToday-CNN datasets. We set $m = 6$ for VSoLSCSum as the same setting in [13] and $m = 4$ for USAToday-CNN because each Web document has 3–4 highlights. Stopwords and links were removed. Table 3 summarizes parameters used in our model.

Table 3. Settings of our model.

Parameter	Value
Batch size	50
Number of epochs	25
Word dimension (Word2Vec [10]) ⁵	25
Hidden size of two MLP layers	20, 1
Learning rate	1.0
Dropout rate	0.5
CNN kernels	$h \in \{1, 2, 3\}$
Subset size k of the second pooling	2
Optimizer	Adadelta
The loss function	Cross-entropy

In training, 15% training data were randomly selected to form a validation set which helps our model to avoid over-fitting.

Evaluation Metrics. Gold-standard references in SoLSCSum and VSoLSCSum, and highlights in USAToday-CNN were used for evaluation by using ROUGE-N F-1 [8] implemented in ROUGE-1.5.5 (N=1, 2).⁶

4 Results and Discussion

This section reports our experiments in three aspects: ROUGE-scores, the analysis of feature contribution, and output observation.

4.1 ROUGE-Scores Observation

CNN-Based Models. We first compared our model to PriorSum, which also uses CNN for ranking sentences [1]. Table 4 reports our comparison. ROUGE-scores show two interesting points. Firstly, as our expectation, our model is consistently better than PriorSum. Improvements come from two factors. (i) Besides directly combining feature maps from tri-grams, we create a subset combination, which improves the representation of each input sentence. (ii) Human knowledge in the form of features enriches vector representation, which benefits in estimating the importance of each sentence and user post. For example, our model is better 2.6% of ROUGE-1 than PriorSum for sentence selection on USAToday-CNN. As mentioned, the small number of training examples on our datasets (see Table 1) challenges PriorSum and our model. Hence, adding features profits the estimation. For user post extraction, the trend is consistent. Our model also surpasses the basic one in all cases. Secondly, our model with features achieves better ROUGE-scores than the model without using features. It is understandable that deep learning learns representation from data; therefore, small training examples challenge such models. In this aspect, additional features from social context improve the performance of our model.

Table 4. Our model vs. PriorSum. RG is ROUGE. Value means our model is significantly better with $p \leq 0.05$ using the pairwise t -test. F stands for features.

Method	SoLSCSum				VSoLSCSum				USAToday-CNN			
	Sentence		Comment		Sentence		Comment		Sentence		Tweet	
	RG-1	RG-2	RG-1	RG-2	RG-1	RG-2	RG-1	RG-2	RG-1	RG-2	RG-1	RG-2
PSum	0.413	0.367	0.211	0.157	0.543	0.447	<u>0.318</u>	<u>0.161</u>	0.214	0.071	0.242	0.082
Ours	<u>0.424</u>	<u>0.380</u>	<u>0.213</u>	0.156	<u>0.550</u>	0.446	0.430	<u>0.253</u>	<u>0.234</u>	0.088	<u>0.246</u>	0.089
Ours-F	0.428	0.386	0.233	0.177	0.554	0.451	0.430	0.290	0.240	<u>0.087</u>	0.250	0.089

Our Model vs. Non-social Context Methods. We next report the comparison between our model and non-social context methods. **Lead- m** selects top m sentences as a summary [11]. **LexRank** uses a stochastic graph-based method for computing relative importance of textual units for extractive summarization [2]. **SVM**⁷ uses a RBF kernel to train a summarizer by using features in

⁶ The parameters are “-c 95 -2 -1 -U -r 1000 -n 2 -w 1.2 -a -s -f B -m”.

⁷ <http://www.csie.ntu.edu.tw/~cjlin/libsvm/>.

[23]. **CRF** employs a set of features in [19] for single document summarization. Table 5 shows ROUGE-scores of these methods.

Table 5. Our model vs. non-social context methods; * is supervised methods; **bold** is the best and *italic* is the second best. Lead- m is not used for user posts. Value means our model is significantly better with $p \leq 0.05$.

Method	SoLSCSum				VSoLSCSum				USAToday-CNN			
	Sentence		Comment		Sentence		Comment		Sentence		Tweet	
	RG-1	RG-2	RG-1	RG-2	RG-1	RG-2	RG-1	RG-2	RG-1	RG-2	RG-1	RG-2
Lead- m	0.345	0.322	—	—	0.495	0.420	—	—	0.249	0.106	—	—
LRank	0.327	0.243	0.210	0.115	0.506	0.432	0.348	0.198	0.251	0.092	0.193	0.068
SVM*	0.325	0.263	0.152	0.089	0.497	0.440	0.374	0.212	0.261	0.106	0.221	0.084
CRF*	0.393	0.379	0.091	0.075	0.422	0.357	0.111	0.062	0.186	0.088	0.190	0.065
Ours*	0.424	0.380	0.213	0.156	0.550	0.446	0.430	0.253	0.234	0.088	0.246	0.089
Ours-F*	0.428	0.386	0.233	0.177	0.554	0.451	0.430	0.290	0.240	0.087	0.250	0.089

ROUGE-scores from Table 5 indicate a consistent trend with Table 4, in which our model is the best in almost all cases, except for sentence selection on USAToday-CNN. For example, our model is better than CRF, which is a very strong baseline on SoLSCSum, e.g. 0.428 vs. 0.393. It is similar to VSoLSCSum, in that two CNN-based methods significantly surpass baselines (values with text are significant with $p \leq 0.05$). This again confirms the efficiency of our model, which takes advantage of CNN for capturing internal features and exploits social context for providing additional useful information.

On USAToday-CNN, on the one hand, SVM is the best, followed by LexRank for sentence selection. It is possible to explain that features used by SVM are appropriate for this dataset. However, on other datasets, SVM does not prove to be efficient. Our model is in the third position even it uses many sophisticated features. This is because all our features are for extraction, but this dataset is of abstraction, which also challenges other advanced methods. For example, ROUGE-scores of RankBoost CCF [21] are lower than Lead- m (see Table 6). This also shows the limitation of CNN in capturing hidden features. On the other hand, for tweet extraction, our methods are the best. The reason is that we enhance vector representation by using a combination of feature maps and integrating social context denoted by features.

Our Model Vs. Social Context Methods. We finally challenged our model with advanced methods. **cc-TAM**⁸ builds a cross-collection topic-aspect modeling for creating a bipartite graph used by co-ranking [3]. **HGRW** is a variation of LexRank named Heterogeneous Graph Random Walk [22]. **RankBoost (RB)** exploits set of local and cross features trained by RankBoost⁹ for training two

⁸ We acknowledge Gao et al., for sharing two parts of code [3].

⁹ <https://people.cs.umass.edu/~vdang/ranklib.html>.

L2R models [21], for sentences and tweets. **SVMRank** is an extension of RB, that the authors added more advanced features [16]. **Voting** bases on three L2R models [15]. Table 6 shows their ROUGE-scores.

Table 6. CNN-based models vs. social context methods.

Method	SoLSCSum				VSoLSCSum				USAToday-CNN			
	Sentence		Comment		Sentence		Comment		Sentence		Tweet	
	RG-1	RG-2	RG-1	RG-2	RG-1	RG-2	RG-1	RG-2	RG-1	RG-2	RG-1	RG-2
cc-TAM	<u>0.306</u>	<u>0.238</u>	<u>0.054</u>	<u>0.022</u>	<u>0.488</u>	<u>0.377</u>	<u>0.301</u>	<u>0.167</u>	0.261	0.074	<i>0.248</i>	0.071
HGRW	<u>0.379</u>	<u>0.204</u>	<u>0.209</u>	0.115	0.570	0.479	0.454	0.298	<i>0.279</i>	<i>0.098</i>	0.242	0.088
RB*	<u>0.360</u>	<u>0.283</u>	<u>0.190</u>	<u>0.098</u>	0.561	0.494	<i>0.471</i>	<i>0.308</i>	0.221	0.070	0.233	<i>0.091</i>
SVMR*	0.381	0.304	0.209	0.122	<i>0.572</i>	<i>0.521</i>	0.482	0.319	0.253	0.084	0.213	0.080
Voting	0.401	0.322	0.223	0.132	0.576	0.523	0.467	0.319	0.287	0.114	0.243	0.095
Ours*	<i>0.424</i>	<i>0.380</i>	<i>0.213</i>	0.156	0.550	0.446	0.430	0.253	0.234	<i>0.088</i>	0.246	0.089
Ours-F*	0.428	0.386	0.233	0.177	0.554	0.451	0.430	0.290	0.240	0.087	0.250	0.089

Results from Table 6 are similar to those in Table 5, in which our model produces promising results. For example, it is competitive on SoLSCSum, and on USAToday-CNN for tweet extraction. Among social context methods, HRGW shows its efficiency, in which it achieves good performance on three datasets. This is because HRGW extends LexRank to utilize the social information of a Web document in an appropriate form. cc-TAM is the second best of ROUGE-1 for tweet selection on USAToday-CNN. Results from unsupervised methods motivate that we can improve their quality to reach the performance of supervised learning ones by using social information in a suitable way. L2R-based models also obtain consistent results. As discussed, it is trained with sophisticated features leading to improvements compared to unsupervised learning models, e.g. cc-TAM. Our model is not the best on VSoLSCSum because of a possible reason that we can not extract some features from this dataset, e.g. indicator words, due to the limitation of tools in Vietnamese.

4.2 Feature Contribution

We observed the contribution of feature groups in our model. To do that, we ran the model with four settings, using: (i) all features, (ii) local features, (iii) user-post features, and (iv) using third-party features. It is possible to observe the influence of each feature; however, due to training time, we leave this observation as a future task. We plot our observation on Figs. 1 and 2.

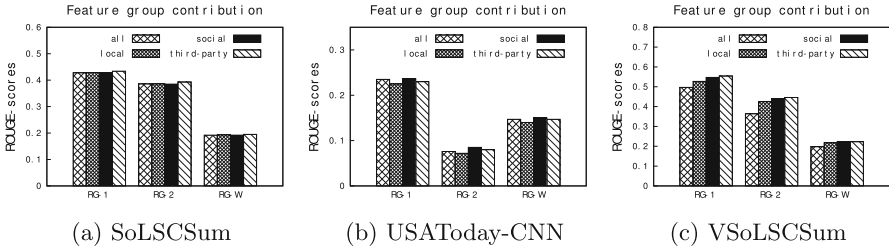


Fig. 1. Feature group contribution of sentence selection.

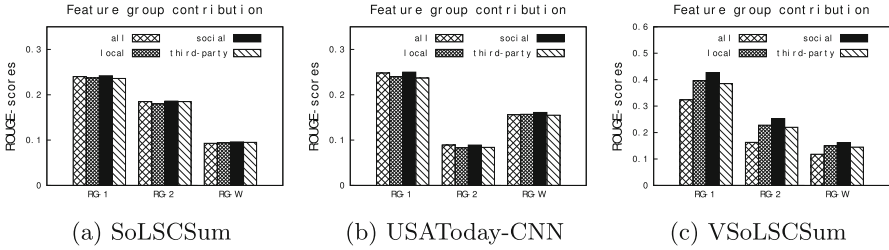


Fig. 2. Feature group contribution of user post extraction.

The general trend on SoLSCSum and USAToday-CNN indicates that: (i) there are small margins among feature groups and (ii) using all and user-post features seems to be more efficient than other groups. This is because they include some good indicators, e.g. sentence length, which can support latent features extracted by CNN. By contrast, the tendency on VSoLSCSum is different, indicating that using all features obtains the worst results. The reason may come from the conflict when combining many features. ROUGE-scores also show that using user-post and third-party features may be appropriate for this dataset.

4.3 Output Observation

Table 7 shows outputs of CNN-based methods extracted from the document “Seconds before crash, passengers knew they were too low” (the Asiana Airlines Flight 214 crash event). We can observe that extracted sentences and tweets provide useful information for readers on this event. This shows the efficiency of two methods in extracting summaries. Their outputs also share common extracted sentences and tweets because they have similar behavior of extraction. This comes from the reason that they employ CNN for learning. On the other hand, they also extract different outputs, indicating that two models produce different summaries even their scores are quite similar.

For sentence selection, PriorSum extracts S2 and S3, which seem to be relevant to the highlights. By reading these sentences, we can partly guess the situation of the event. However, it selects S1, containing information in the past, which may support the main story but it does not directly relate to the event.

This is similar to S4, which shows the opinion of Hayes-White. By contrast, outputs of our model may be closer to the highlights than those of PriorSum. For example, we know the total number of passengers (in S3) or what happened with flight recorders (in S2). They provide richer information than PriorSum.

Table 7. A summary example of the document 14th on USAToday-CNN.

Highlights
Flight recorders have been found, the NTSB says Asiana identifies the two 16-year-old girls killed in the crash 182 people were hospitalized, while 123 were uninjured Passengers say the plane’s rear struck the edge of the runway
Outputs of PriorSum
Sentence selection
S1: In 1993, a crash near South Korea’s Mokpo Airport killed 68 of the 116 people on board S2: The flight recorders from the plane have been recovered and are on the way to Washington, the NTSB said Sunday S3: The Boeing 737-500 went down in poor weather as the plane was attempting its third landing, the Aviation Safety Network said S4: “We’re lucky there hasn’t been a greater loss of life,” San Francisco Fire Chief Joanne Hayes-White said
Tweet extraction
T1: Seconds before crash, passengers knew they were too low - Asiana Airlines Flight 214 was seconds away from landing T2: NTSB says Asiana Airlines Flight 214 flight recorders have been found. - @CNN T3: That had to be scary! ”@cnnbrk: Flight recorders recovered from San Francisco crash site, NTSB says T4: Seconds before crash, passengers knew they were too low Look these pics! A terrible plane accident in San Francisco
Outputs of our model with features
Sentence selection
S1: Perhaps one of the reasons so many people survived Saturday’s crash was because the Boeing 777 is built so that everybody can get off the plane within 90s, even if half the doors are inoperable S2: The flight recorders from the plane have been recovered and are on the way to Washington, the NTSB said Sunday S3: Once the plane fell short of the runway, passengers found themselves on a roller coaster S4: Asiana Airlines Flight 214 was seconds away from landing when the passengers sensed something horribly amiss
Tweet extraction
T1: NTSB says Asiana Airlines Flight 214 flight recorders have been found. - @CNN T2: That had to be scary! ”@cnnbrk: Flight recorders recovered from San Francisco crash site, NTSB says T3: 2 teens killed in San Francisco plane crash; 182 hospitalized - CNN: ABC News2 teens killed in San Francisco T4: Seconds before crash, passengers knew they were too low - Asiana Airlines Flight 214 was seconds away from landing

For tweet extraction, two methods output valuable tweets, which include important information. For example, T2 of PriorSum or T1 of our model completely match with the first highlight. Our model selects two very important tweets: T1 and T3. While T1 is similar to PriorSum, T3 provides critical information, which totally matches with the second and the third highlight. It includes the status of two teens (“*killed*”), the name of the event (“*San Francisco plane crash*”), and the number of victims (“*182 hospitalized*”). Extracted tweets from our model can cover almost important information from the highlights. Two methods also share some common tweets due to the use of CNN.

5 Conclusion

This paper presents a CNN-based model for summarizing Web documents by exploiting their social context. The model provides a way to enhance the representation of sentences and user posts by combining hidden and external features. The enhancement benefits the learning process to capture more important information. We carefully conduct experiments on three datasets in two languages, English and Vietnamese. Promising results confirm that our model can take advantage of social context to improve the quality of sentence ranking, which benefits to extract high-quality summaries. Applying the model to sentence and highlight extraction tasks of single documents shows that it can be viable alternative to extraction-based systems.

Future work will investigate features to reveal the role of each indicator. Our model should integrate LSTM to exploit the contextual sentence relations.

Acknowledgment. This work was supported by JSPS KAKENHI grant numbers JP15K16048 and JP15K12094, and Center for Research and Applications in Science and Technology, Hung Yen University of Technology and Education, under the grant number UTEHY.T026.P1718.04.

References

1. Cao, Z., Wei, F., Li, S., Li, W., Zhou, M., Wang, H.: Learning summary prior representation for extractive summarization. In: ACL, vol. 2, pp. 829–833 (2015)
2. Erkan, G., Radev, D.R.: Lexrank: graph-based lexical centrality as salience in text summarization. *J. Artif. Intell. Res.* **22**, 457–479 (2004)
3. Gao, W., Li, P., Darwish, K.: Joint topic modeling for event summarization across news and social media streams. In: CIKM, pp. 1173–1182 (2012)
4. Hu, M., Sun, A., Lim, E.P.: Comments-oriented document summarization: understanding document with readers’ feedback. In: SIGIR, pp. 291–298 (2008)
5. Kim, Y.: Convolutional neural networks for sentence classification. In: EMNLP, pp. 1746–1751 (2014)
6. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. *Proc. IEEE* **86**(11), 2278–2324 (1998)
7. Li, C., Wei, Z., Liu, Y., Jin, Y., Huang, F.: Using relevant public posts to enhance news article summarization. In: Coling, pp. 557–566 (2016)

8. Lin, C.Y., Hovy, E.H.: Automatic evaluation of summaries using n-gram co-occurrence statistics. In: HLT-NAACL, pp. 71–78 (2003)
9. Lu, Y., Zhai, X., Sundaresan, N.: Rated aspect summarization of short comments. In: WWW, pp. 131–140 (2009)
10. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J.: Distributed representations of words and phrases and their compositionality. In: NIPS, pp. 3111–3119 (2013)
11. Nenkova, A.: Automatic text summarization of newswire: lessons learned from the document understanding conference. In: AAAI, pp. 1436–1441 (2005)
12. Nguyen, M.T., Cuong, T.V., Hoai, N.X., Nguyen, M.L.: Utilizing user posts to enrich web document summarization with matrix co-factorization. In: The Eight International Symposium on Information and Communication Technology (SoICT), pp. 70–77 (2017)
13. Nguyen, M.T., Lai, V.D., Do, P.K., Tran, D.V., Nguyen, M.L.: Vsolscsum: building a Vietnamese sentence-comment dataset for social context summarization. In: The 12th Workshop on Asian Language Resources, pp. 38–48 (2016)
14. Nguyen, M.T., Tran, C.X., Tran, D.V., Nguyen, M.L.: Solscsum: a linked sentence-comment dataset for social context summarization. In: CIKM, pp. 2409–2412 (2016)
15. Nguyen, M.-T., Tran, D.-V., Nguyen, L.-M.: Social context summarization using user-generated content and third-party sources. *Knowl.-Based Syst.* **144**, 51–64 (2018)
16. Nguyen, M.T., Tran, D.V., Tran, C.X., Nguyen, M.L.: Learning to summarize web documents using social information. In: ICTAI, pp. 619–626 (2016)
17. Nguyen, M.T., Tran, D.V., Tran, C.X., Nguyen, M.L.: Summarizing web documents using sequence labeling with user-generated content and third-party sources. In: NLDB, pp. 454–467 (2017)
18. Nguyen, M.-T., Tran, D.-V., Tran, X.-C., Nguyen, L.-M.: Exploiting user-generated content to enrich web document summarization. *Int. J. Artif. Intell. Tools* **26**(5), 1–26 (2017)
19. Shen, D., Sun, J.T., Li, H., Yang, Q., Chen, Z.: Document summarization using conditional random fields. In: IJCAI, pp. 2862–2867 (2007)
20. Sun, J.T., Shen, D., Zeng, H.J., Yang, Q., Lu, Y., Chen, Z.: Web-page summarization using clickthrough data. In: SIGIR, pp. 194–201 (2005)
21. Wei, Z., Gao, W.: Utilizing microblogs for automatic news highlights extraction. In: Coling, pp. 872–883 (2014)
22. Wei, Z., Gao, W.: Gibberish, assistant, or master?: using tweets linking to news for extractive single-document summarization. In: SIGIR, pp. 1003–1006 (2015)
23. Yang, Z., Cai, K., Tang, J., Zhang, L., Su, Z., Li, J.: Social context summarization. In: SIGIR, pp. 255–264 (2011)



Towards Event Timeline Generation from Vietnamese News

Van-Chung Vu¹, Thi-Thanh Ha^{1,2}, and Kiem-Hieu Nguyen^{1(✉)}

¹ School of Information and Communication Technology,
Hanoi University of Science and Technology, Hanoi, Vietnam
htthanh@ictu.edu.vn, hieunk@soict.hust.edu.vn

² Thai Nguyen University of Information and Communication Technology,
Thai Nguyen, Vietnam

Abstract. Event timeline generation is an active research area in many languages. In this paper, we describe our attempts towards event timeline generation from Vietnamese newswire documents based on the work in [1]. Our main contributions are: *i*) To our knowledge, we are the first to tackle the problem for Vietnamese language. *ii*) Experiments were conducted on a large-scale corpus from 145 popular online news agents in the period from 2007 to 2017 which provides extensive redundant information on various themes. *iii*) We manually built a dataset of 17 timelines for evaluation (The dataset could be downloaded at <http://is.hust.edu.vn/~hieunk/resources/vntimeline17.zip>). Experimental results show that the proposed method works reasonably well for Vietnamese; and using n-grams on unsegmented texts achieves comparable performance to using word segmentation.

Keywords: N-gram model · Timeline generation · Temporal analysis · Event extraction · Vietnamese

1 Introduction

Text summarization is an active research area in NLP [2]. Event timeline generation could be considered as a branch of text summarization which focuses on summarizing thematic events [1]. It is closely related to query-focused multidocument summarization [3] and topic detection and tracking [4]. The differences are two-fold: Firstly, it works on event-centric documents; Secondly, it considers not only textual information but also temporal information when summarizing. It eventually distinguishes from timeline generation for an individual entity [5].

By *thematic event*, we are referring to a collection of events relevant to a common theme. For example, the thematic event “Vụ con ruồi trong chai Number 1 (The fly in Number 1)”¹ started with an event “A customer found a fly

¹ For convenience of readers who are not familiar with Vietnamese, we are going to use the English translation instead of the original texts in Vietnamese in the rest of the paper.

inside an unopened Number 1 bottle” and continued with another event such as “He then contacted Tan Hiep Phat company to blackmail 1 billion VND”, etc. Our task is to generate a timeline of the most salient events relevant to a thematic event as in Fig. 1. Following previous works on event timeline generation, we treat an event as a pair of timestamp and an event description. For instance, (2014/12/03, “A customer found a fly inside an unopened Number 1 bottle”) is an event. We are not going to use event details such as triggers and participants as in Message Understanding Conference and Automatic Content Extraction (and its subsequent Knowledge Base Population) [6–8].

When reporting on a thematic event, such as a politic scandal or a disaster, different journalists tend to agree on salient events and story developments. Such agreement is consequently reflected on redundancy between news sources. We aim at leveraging this redundancy to detect salient events.

The task has been studied in other languages like English and French [9]. However, to our knowledge, there has been no study in Vietnamese so far although there is a large and rapidly increasing volume of event-related newswire contents in Vietnamese.

In this paper, we present our approach towards generating event timelines on Vietnamese newswire documents. Our contributions are three-fold: Firstly, to our knowledge, we are the first to tackle this problem in Vietnamese. Secondly, as redundancy from various sources is crucial in order to judge importance of information related to an event, we have gathered a large-scale corpus of Vietnamese news from 145 popular online news agencies in Vietnam in the period from 2007 to 2017. As illustrated by experimental results, such corpus is adequate for generating timelines of various events. Last but not least, we have manually created a dataset of 17 timelines for evaluation. The dataset consists of various events happening in Vietnam as well as world-wide. Experimental results on the dataset show that the models based on n-grams are on par with the word-based model.

The rest of the paper is organized as follows: Sect. 2 briefly introduces related work; Sect. 3 describes our proposed method; Sect. 4 demonstrates evaluation results; The paper is concluded in Sect. 5.

2 Related Work

Most works on event timeline generation require redundancy from data. [10] proposed a framework for extensive temporal analysis and used redundant data and machine learning to detect salient dates of thematic events. Following this direction, [1] presented a Maximal Marginal Relevance-like reranking algorithm based on both temporal and thematic clustering [11]. In another approach, [12] proposed a joint graphical model for the problem. The problem has been also tackled in other languages such as French [9]. In an attempt to improve evaluation methodology, [13] proposed a metric based on so-called deep semantic units.

In this work, we aim at applying the approach as described in [1] to a new language, i.e. Vietnamese in our case. Therefore, we follow the essential parts

Vụ con ruồi trong chai nước Number 1

- 2014/12/03: Ông Võ Văn Minh (35 tuổi, Tiền Giang) lấy chai number one bán cho khách hàng và phát hiện con ruồi trong chai chưa khui nắp.
- 2014/12/05: Ông Minh gọi điện cho công ty TNHH thương mại dịch vụ để nghị công ty đưa 1 tỉ đồng nếu không đưa sẽ kiện ra hội đồng bảo vệ người tiêu dùng.
- 2014/12/06: Nhân viên Tân Hiệp Phát đến lần 1 không đưa tiền mà chỉ tặng sản phẩm và xin lại chai nước.
- 2014/12/16: Nhân Viên Tân Hiệp Phát đến lần 2, lập biên bản ghi nhận. Ông Minh đề nghị đưa 500 triệu
- 2015/01/20: Nhân viên Tân Hiệp Phát đến lần 3. Ông Minh đề nghị 500 triệu
- 2015/01/27: Nhân viên Tân Hiệp Phát giao 500 triệu và ông Minh đưa chai nước, nhận tiền và bị công an bắt. Ông Minh bị khởi tố cưỡng đoạt tài sản.
- 2015/02/13: Thanh tra số y tế tỉnh Bình Dương thanh tra dây truyền của Tân Hiệp Phát.
- 2015/02/14: Kết luận Thanh tra không vi phạm.
- 2015/05/27: Công an gia hạn tạm giam ông Minh, Kết quả giám định nắp chai không còn nguyên.
- 2015/09/08: Công an kết luận ông Minh phạm tội cưỡng đoạt tài sản và đề nghị truy tố.
- 2015/10/09: Ra cáo trạng và chuyển hồ sơ sang tòa.
- 2015/12/17: Xét xử sơ thẩm, Minh bị lĩnh 7 năm tù
- 2015/12/18: Tòa án ND tỉnh Tiền Giang tuyên án 7 năm tù với Nguyễn Văn Minh
- 2015/12/19: Phó Tổng giám đốc xin lỗi người tiêu dùng
- 2016/01/21: Người dân phát hiện ruồi trong chai Number 1 và được đưa ra hội bảo vệ người tiêu dùng.
- 2016/12/23: Tổng giám đốc xin lỗi người tiêu dùng

Fig. 1. A timeline of “The fly in Number 1” written by journalists.

of the method including document acquisition, temporal analysis, event ranking and selection.

Applying to a resource-scarce language like Vietnamese is not trivial. The main issues are:

1. Acquiring a large, redundant news corpus over a long period.
2. Processing temporal analysis.
3. Dealing with unsegmented texts, i.e. texts in which word delimiters are ambiguous. This is a specific problems in some Asian languages like Chinese, Japanese, and Vietnamese.
4. Creating reference timelines for evaluation.

We will discuss in more details these issues and our proposed solutions in subsequent sections.

3 Our Event Timeline Generation Method

In the first phase, from a large, redundant news corpus, temporal analysis is processed on the whole corpus. That is to say, whenever there is a temporal expression, it will be detected and will be normalized. Sentences containing at least one temporal expression are then collected. We use Lucene² search engine to index events as pairs of date and description sentences. Moreover, sentences containing the same normalized time will be gathered into a cluster. We use Lucene ranking algorithm to rank and select salient events for an input query and to generate the final timeline. In this paper, we follow previous works to

² <https://lucene.apache.org>.

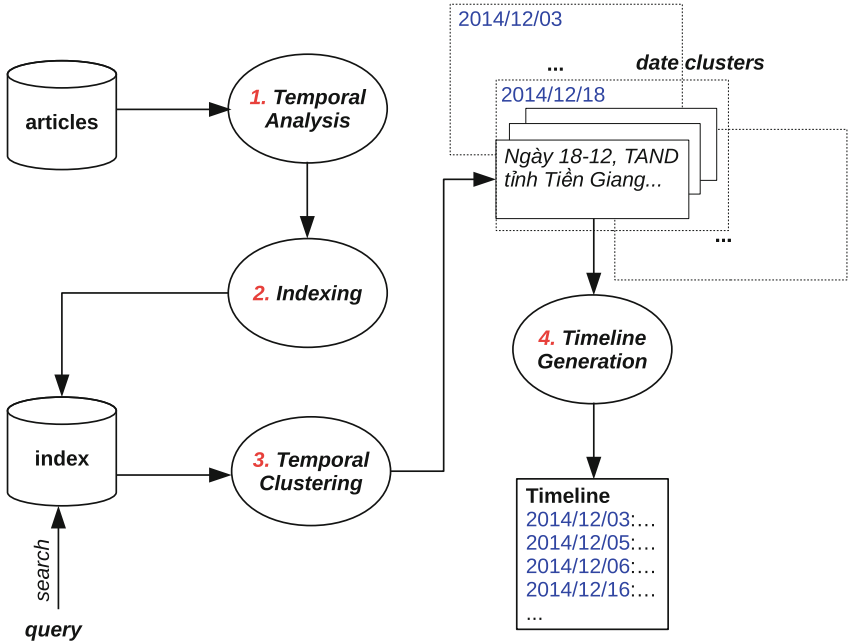


Fig. 2. Our event timeline generation method

use date as temporal interval. Each date cluster hence consists of all sentences belong to an individual date. Our method is demonstrated in Fig. 2. Readers could refer to [1] for more details on the original method.

In Sect. 3.1, we describe our news corpus. Sections 3.2 and 3.3 present temporal analysis and indexing, respectively. Section 3.4 is dedicated to temporal clustering. Timeline generation is discussed in Sect. 3.5.

3.1 The News Corpus

A large, redundant corpus is crucial in this work. To obtain such a corpus in Vietnamese, we first surveyed popular online news agencies in Vietnamese. From that, we selected 145 most popular sites. Articles from these sites between the year of 2007 and 2017 were gathered, resulting in totally 3.8M articles. HTML documents were parsed and all contents and meta-data such as title, document creation time (DCT), tags, URL were stored in XML format. Note that DCT is required for temporal analysis. Most of the articles comes from dominant agencies in Vietnam such as VnExpress, TuoiTre, DanTri, Vietnamnet. The content varies from social-economics, politics to hi-tech, entertainments, world wide events.

3.2 Phase 1: Temporal Analysis

Temporal analysis consists of detecting and normalizing temporal expressions in texts. It is shown in [10] that only 7% of temporal expressions in texts are absolute dates (i.e. with date, month, and year), the rest DCT-related dates require normalization. For example, in the sentence “Ngày 18–12, TAND tỉnh Tiền Giang tuyên Minh 7 năm tù vì tội Cường đoạt tài sản”, we need to detect “Ngày 18–12” and normalize it as 2004/12/18. Other expressions such as “ngày hôm qua” (yesterday) or “Thứ Sáu tuần trước” (last Friday) are even more challenging.

In our experiments, we used HeidelTime, a multilingual temporal analyzer which supports Vietnamese [14]. To our knowledge, it is the only temporal analyzer to date for Vietnamese. It uses JVNTextPro³ for word segmentation and part-of-speech tagging as prerequisite for temporal analysis.

After temporal analysis, we remove all sentences without temporal expressions as well as sentences with normalized temporal expressions that are incomplete (e.g. 2015/10/xx where date is missing). These sentences are indexed using Lucene search engine, each one as a *document*, resulting in an index of totally 11.6M documents. Statistics of the corpus is shown in Table 1.

Table 1. Corpus statistics.

Item	Number
Sites	145
Articles	3,801,523
Sentences	54,932,191
Sentences with temporal expressions	11,596,796

3.3 Phase 2: Indexing

Vietnamese texts don’t have explicit word boundaries. Spaces, which are natural word boundaries in languages such as English, only serve as boundary between syllables in Vietnamese. Word segmentation is required for detecting word boundaries, i.e. deciding whether or not a space is a word boundary or is inside a multi-syllable word. Word segmentation is useful for downstream problems like syntactic parsing and semantic parsing. It has been shown in [15] that using n-grams is comparable to using words for information retrieval of Chinese texts.

To investigate the impact of word segmentation, in our experiments, we built three indices using unigram, bigram, and word. For example, “truy hỏi (retrieval) thông tin (information)” is indexed as {‘truy’, ‘hỏi’, ‘thông’, ‘tin’}, {‘truy hỏi’, ‘hỏi thông’, ‘thông tin’}, and {‘truy hỏi’, ‘thông tin’} using unigram, bigram, and word, respectively. VnTokenizer [16] was used for word segmentation.

³ <http://jvntextpro.sourceforge.net/>.

3.4 Phase 3: Temporal Clustering

When a query, such as “The fly in Number 1”, is fed into the Lucene index, it will return relevant documents to the query. In our experiments, we used the built-in tf-idf scoring function of Lucene, and limited to 10K documents for each query. All events having the same date value are gathered into a *date cluster*. Date cluster is a central notion in our method. If a thematic event lasts over a long period, the dates on which many events happen tend to be more salient than the others. Moreover, within a date cluster, not all the events are equivalently important. Salient events take an essential part in the whole story. Marginal events, for instance, could be some kind of reactions, or could describe minor details. The most important events are duplicated in several descriptions because they are reported by many news sources.

3.5 Phase 4: Timeline Generation

Timeline generation consists of selecting the most salient dates and selecting the most salient event in each date. Saliency score of a date d is the accumulation of Lucene scores⁴ of all events e in d regarding a query q :

$$saliency(d) = \sum_{e \in d} score_{Lucene}(e, q) \quad (1)$$

Lucene score of an event reflects the relevance of its description to the query.

For event ranking inside a date d , we simply select the one with the highest Lucene score as the representative event of d :

$$\arg \max_{e \in d} score_{Lucene}(e, q) \quad (2)$$

The resulting timeline has K events and could be ordered by saliency or chronology. The number of events K could be varied to show only salient events or to get more details.

4 Evaluations

In this section, we describe the creation of a dataset of 17 reference timelines in Sect. 4.1. Our evaluation follows the work in [1]. Temporal contents of the timeline are evaluated by salient date detection (Sect. 4.2). Textual contents are evaluated by text summarization (Sect. 4.3).

4.1 Creation of Reference Timelines

We first investigated timelines written in Vietnamese by journalists. There were not many such timelines at the moment we conducted the experiments. Timelines describing events out of the period 2007 and 2017, which are not available in

⁴ <https://lucene.apache.org/core/3.6.0/scoring.html>.

our corpus, are ignored. Our final dataset contains 17 timelines, for both events happening in Vietnam (e.g. “Airplane crashing in Hoa Lac”) and worldwide (e.g. “Missing plane MH370”) as shown in Table 2.

4.2 Evaluating Salient Date Detection

Table 2. Evaluating salient date detection.

No Query	UNIGRAM	BIGRAM	WORD
1 Nguyễn Thanh Chấn (Nguyen Thanh Chan)	26.4	26.1	24.4
2 Cá chết hàng loạt ở miền Trung (Dead fish spreading in Middle Vietnam)	41.9	41.3	43.5
3 Cà phê “Xin Chào” (“Hello” café)	57.9	60.2	54.9
4 Cao Toàn Mỹ - Trương Hồ Phương Nga (Cao Toan My - Truong Ho Phuong Nga)	13.1	16.7	11.7
5 Cháy karaoke Cầu Giấy (Cau Giay karaoke on fire)	44.7	47.9	43.8
6 Leo thang căng thẳng trên bán đảo Triều Tiên (Escalating tensions in Korea Peninsula)	10.7	14.8	12.4
7 Máy bay mất tích MH370 (Missing plane MH370)	34.4	34.1	38.2
8 Philippin - Trung Quốc (Philippines - China)	60.6	49.1	54.9
9 Máy bay rơi Hòa Lạc (Airplane crashing in Hoa Lac)	3.1	2.0	1.3
10 Thẩm mỹ viện Cát Tường (Cat Tuong beauty saloon)	48.0	48.4	43.7
11 Giàn khoan 981 (HD-981 Oil Rig)	70.6	74.2	73.8
12 Bê bối của Tổng Thống Hàn Quốc (South Korean president scandal)	48.7	49.2	51.1
13 Vụ con ruồi trong chai Number 1 (The fly in Number 1)	55.6	52.3	54.9
14 Đắm phà Sewol (Sewol sinking ferry)	48.6	45.1	47.9
15 Đông Tâm, Mỹ Đức (Dong Tam, My Duc)	11.0	13.8	7.2
16 Huỳnh Thị Huyền Như (Huynh Thi Huyen Nhu)	30.2	34.2	18.8
17 Trịnh Xuân Thanh (Trinh Xuan Thanh)	41.3	54.5	53.1
MAP	38.1	39.0	37.4

We used Mean Average Precision (MAP) to evaluate salient date detection on three systems, each uses one of the three indices: unigram, bigram, and word. For each query, the ranked list of all the dates returned by a system according

to Equation (2) is compared against the dates in the reference timeline. If a date in the reference timeline is not retrieved by the system, its average precision is counted as zero.

$$MAP = \frac{1}{N} \sum_{j=1}^N \left(\frac{1}{Q_j} \sum_{i=1}^{Q_j} P(date_i) \right) \quad (3)$$

Q_j : number of relevant dates for query j

N : number of queries

$P(date_i)$: precision at i th relevant date

As shown in Table 2, UNIGRAM and BIGRAM perform slightly better than WORD. Performance varies across the timelines. The main reasons for poor performance are: An event lasts for too long (“Cao Toan My - Truong Ho Phuong Nga” lasts for three years); The event lasts only in a few days and there are many events of the same type happening in the same time but in other locations (“Airplane crashing in Hoa Lac”); There are not many news about the event, such as the riot in Dong Tam, My Duc; The event lasts for too long but the reference timeline only covers a specific period (The timeline about “Escalating tensions in Korea Peninsula” only covers events in 2017). In fact, one interesting aspect of our method is that when a user want to focus on a particular period, he could limit search range accordingly. For example, one could zoom in for events about Korea Peninsula in 2017. Implementing this feature in Lucene is straightforward.

4.3 Evaluating Text Summarization

For each query, top K dates from the system are selected. The most relevant sentence in each date is then extracted. The final timeline contains K sentences. Here, K is the number of dates in the reference timeline. We use ROUGE metric [17] to evaluate generated timelines against reference timelines.

Table 3. Evaluating text summarization.

System	ROUGE-1	ROUGE-2	ROUGE-L
UNIGRAM	39.0	16.4	26.3
BIGRAM	38.3	16.1	25.5
WORD	40.1	15.3	24.9

Table 3 again demonstrates that UNIGRAM and BIGRAM perform equally to WORD, while requiring no word segmentation. On the other hands, the results are fluctuated. This is probably because we only have 17 queries and there is only one timeline per query, that could reflect subjectivity in timeline contents.

5 Conclusions and Future Works

This paper presents a timeline generation system for Vietnamese. The experiments were conducted on a large newswire corpus of articles in various domains. We empirically show that using unigrams and bigrams produces timelines of comparable quality without word segmentation.

There are more rooms for improvement. We could applying event clustering to highlight important events and event reranking for diversity as in [1]. Unlike their work, we don't have keywords in reference timeline. Therefore, putting only event *titles*, sometimes too short or too ambiguous, into Lucene could return in many irrelevant results. Query expansion technique could be a useful and feasible solution. Moreover, we are going to expand the reference dataset that has multiple timelines per query for more robust evaluation. One possibility is using existed dates from timelines written in English about world-wide events for evaluating salient date detection, and further manually translating those timelines into Vietnamese for evaluating text summarization. Another direction for enhancement is improving temporal analysis for Vietnamese. TIME named entities are not only crucial for timeline generation but they are also important in other tasks such as event extraction and knowledge base population.

Acknowledgments. We would like to thank Vccorp for kindly supporting us on conducting the experiments.

References

1. Nguyen, K.H., Tannier, X., Moriceau, V.: Ranking multidocument event descriptions for building thematic timelines. In: Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, Dublin, Ireland, Dublin City University and Association for Computational Linguistics, pp. 1208–1217 (2014)
2. Mani, I.: Advances in Automatic Text Summarization. MIT Press, Cambridge (1999)
3. Goldstein, J., Mittal, V., Carbonell, J., Kantrowitz, M.: Multi-document summarization by sentence extraction. In: Proceedings of the 2000 NAACL-ANLPWorkshop on Automatic Summarization, NAACL-ANLP-AutoSum 2000, Stroudsburg, PA, USA, vol. 4, pp. 40–48. Association for Computational Linguistics (2000)
4. Allan, J.: Topic Detection and Tracking, pp. 1–16. Kluwer Academic Publishers, Norwell (2002)
5. Li, J., Cardie, C.: Timeline generation: tracking individuals on twitter. In: Proceedings of the 23rd International Conference on World Wide Web, WWW 2014, New York, NY, USA, pp. 643–652. ACM (2014)
6. Grishman, R., Sundheim, B.: Message understanding conference-6: A brief history. In: Proceedings of the 16th Conference on Computational Linguistics, COLING 1996, Stroudsburg, PA, USA, vol. 1, pp. 466–471. Association for Computational Linguistics (1996)

7. Doddington, G., Mitchell, A., Przybocki, M., Ramshaw, L., Strassel, S., Weischedel, R.: The automatic content extraction (ace) program tasks, data, and evaluation. In: Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC-2004), Lisbon, Portugal, European Language Resources Association (ELRA) ACL Anthology Identifier: L04-1011 (2004)
8. Ji, H., Grishman, R.: Knowledge base population: Successful approaches and challenges. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, HLT 2011, Stroudsburg, PA, USA, vol. 1, pp. 1148–1158. Association for Computational Linguistics (2011)
9. Battistelli, D., Charnois, T., Minel, J.L., Teissèdre, C.: Detecting salient events in large corpora by a combination of NLP and data mining techniques. In: Conference on Intelligent Text Processing and Computational Linguistics, Samos, Greece, vol. 17, pp. 229–237 (2013)
10. Kessler, R., Tannier, X., Hagège, C., Moriceau, V., Bittar, A.: Finding salient dates for building thematic timelines. In: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, Jeju Island, Korea, vol. 1: Long Papers, pp. 730–739. Association for Computational Linguistics (2012)
11. Carbonell, J., Goldstein, J.: The use of mmr, diversity-based reranking for reordering documents and producing summaries. In: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 1998, New York, NY, USA, pp. 335–336. ACM (1998)
12. Tran, G., Herder, E., Markert, K.: Joint graphical models for date selection in timeline summarization. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, Beijing, China, vol. 1: Long Papers, pp. 1598–1607. Association for Computational Linguistics (2015)
13. Bauer, S., Teufel, S.: A methodology for evaluating timeline generation algorithms based on deep semantic units. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, Beijing, China, vol. 2: Short Papers, pp. 834–839. Association for Computational Linguistics (2015)
14. Strötgen, J., Gertz, M.: Heideitime: High quality rule-based extraction and normalization of temporal expressions. In: Proceedings of the 5th International Workshop on Semantic Evaluation, SemEval 2010, Stroudsburg, PA, USA, pp. 321–324. Association for Computational Linguistics (2010)
15. Nie, J.Y., Gao, J., Zhang, J., Zhou, M.: On the use of words and n-grams for Chinese information retrieval. In: Proceedings of the Fifth International Workshop on Information Retrieval with Asian Languages, IRAL 2000, New York, NY, USA, pp. 141–148. ACM (2000)
16. Leporati, A., Martín-Vide, C., Shapira, D., Zandron, C. (eds.): LATA 2021. LNCS, vol. 12638. Springer, Cham (2021). <https://doi.org/10.1007/978-3-030-68195-1>
17. Lin, C.Y.: Rouge: a package for automatic evaluation of summaries. In: Marie-Francine Moens, S.S. (ed.) Text Summarization Branches Out: Proceedings of the ACL-04 Workshop, Barcelona, Spain, pp. 74–81. Association for Computational Linguistics (2004)



An Abstractive Text Summarization Using Recurrent Neural Network

Dipanwita Debnath¹(✉), Partha Pakray¹, Ranjita Das¹,
and Alexander Gelbukh²

¹ Department of Computer Science and Engineering,
National Institute of Technology Mizoram, Aizawl, India
{ddeb Nath.nita, parthapakray, ranjita.nitm}@gmail.com

² Center for Computing Research (CIC), Instituto Politécnico Nacional (IPN),
Mexico City, Mexico
gelbukh@cic.ipn.mx

Abstract. With the accelerated advancement of technology and massive content surging over the Internet, it has become an arduous task to abstract the information efficiently. However, automatic text summarization provides an acceptable means for fast procurement of such information in the form of a summary through compression and refinement. Abstractive text summarization, in particular, builds an internal semantic representation of the text and uses natural language generation techniques to create summaries closer to human-generated summaries. This paper uses Long Short Term Memory (LSTM) based Recurrent Neural Network to generate comprehensive abstractive summaries. To train LSTM based model requires a corpus having a significant number of instances containing parallel running article and summary pairs. For this purpose, we have used various news corpus, namely DUC 2003, DUC 2004 and Gigaword corpus, after eliminating the noise and other irrelevant data. Experiments and analyses of this work are performed on a subset of these whole corpora and evaluated using ROUGE evaluation. The experimental result verifies the accuracy and validity of the proposed system.

Keywords: Abstractive text summarization · OpenNMT · ROUGE · Long short term memory · Recurrent Neural Network

1 Introduction

Text Summarization is the process of shortening a text article to create a comprehensive summary that captures the key idea of the article. Text Summarization is a highly challenging job despite the long history of research. It gained widespread interest due to the overwhelming amount of textual information available over the Internet. By extracting key ideas and creating comprehensive summaries,

with the help of a machine, it is possible to assess whether or not a text is worth reading. Text summarization methods are broadly divided into two groups, i.e., abstractive and extractive [14]. The essence of the extractive text summarization is, it is a selection problem, i.e. selecting the best-featured text using machine learning techniques that can represent the main article [12]. The word in the summary generated by extractive summarization is a subset of words of the main article. In contrast, abstractive summarization is a text generation process that requires a deep semantic and discourse understanding of the text. For example, we read a story, understand it and wrote its summary in our language. ATS understands the article and produces the summary using its vocabulary. Hence the summary words are not a subset of article words.

Conventional approaches of summarization have served the purpose for years. However, their underlying demerits and increased aspiration for perfection in summarization enforce the exploration of competent emerging techniques. Hence, we used Open Neural Machine Translation (OpenNMT)¹, a most successful and proficient approach to neural machine-based translation, incorporates use of exhaustively trained large neural networks in summarization process [13] [11]. OpenNMT is successful to a large extent in machine language translation and speech recognition. It is a complete platform with pre-processing, training, testing module. OpenNMT based text summarization model maps the input sequence of the source article to the output sequence of the target summary of the article in the training module. It can further generate similar target summaries from a given test article. Text summarization is a many to many sequence problem as the input and its corresponding summaries are not of the same length, also, the size of the text is larger, unlike machine translation which is of similar length input-output pairs

In neural network-based summarization, the encoder encodes the source article, one token at a time, uses the LSTM and stores the entire encoding data in its last hidden state [13]. Encoded output is fed to the decoder for summary prediction. An improvement in summary quality has been observed by the use of LSTM based Recurrent Neural Networks in place of convolution neural networks ? [11]. However, Comprehensibility, adequacy and fluency of system generated summaries are primarily determined by the implementation approach used for the decoder. For this reason, attention incorporated, LSTM-based RNN is used for designing encoder and decoder of training and testing module.

We have collected a large amount of data having parallel running source and summary pairs and prepared them for training, validation and testing purposes. We have comprehensively analyzed the performance of our OpenNMT based summarization system by varying the dataset and other neural machine parameters. For summary evaluation, we have used Recall-Oriented Understudy for Gisting Evaluation (ROUGE) [8]. It includes measures to automatically determine system-generated summaries' quality by comparing them to other target summaries created by humans of the same source article.

¹ <http://opennmt.net>.

The rest of the paper is organized as follows: Overview of text summarization and related work is presented in Sect. 2. In Sect. 3, the detailed system architecture is described. In Sect. 4. We discussed the corpora and experimental setups used. Section 5 describes system results and their comprehensive analysis. Finally, Sect. 6 concludes the paper with some insights into future work.

2 Background and Related Work

In the field of text summarization, research has been going on for decades through a wide range of past work are performed using extractive methods [?]. At the beginning of 1958, frequency-based summarization was introduced where the frequency is assigned to each word, and top-scored sentences formed the summary [9]. On intuition, meaningful sentences are located at a specific position, such as the beginning or end of a paragraph. Preference to sentence location along with the frequency-based sentence scoring criteria is added by [1]. Headline words similarity and clue word feature is added to this high-frequency content words and sentence location feature [4] in the year 1969. Most of the conventional summarization systems use extractive approaches based on human-engineered features. The abstraction-based models primarily provide the summary by sentence compression and reformulation, which requires complex linguistic processing [3]?? and requires more effort as compared to an extractive summary but provides a summary similar to human-created.

Abstractive Text summarization using neural networks gained widespread interest because of its noticeable performance. Though neural machine translation showed its emergence in 1987 after several modifications and research, it again gained its popularity in 2013. Long Short Term Memory (LSTM) based Recurrent Neural Networks (RNN) were used in the paper [2, 13] to encode a variable-length source sentence into a fixed-length vector and to decode the vector into a variable-length target sentence. A similar 2-layer LSTM (Long Short Term Memory) based RNN encoder-decoder Neural Machine Translation System facilitating encoding of variable length source sentence into fixed-length vector and decoding of fixed length vectors to get target sentence [6] can be used for Abstractive Text summarization as it can read articles of variable length and generate summaries based on learning mechanism of RRN. Abstraction based summarization on news corpus, named them as ABS and ABS+, achieved the state of the art since it outperformed all previously published models on summarization [11] based on both abstractive and extractive, neural or non-neural systems.

Recently, several papers have proposed the use of neural networks [2, 11, 13]?? for machine translation and neural network-based summarization, which motivated us to use the same concept for text summarization. This neural machine translation approach typically consists of the input layer, several hidden layers and an output layer. Every layer encodes a source sentence, corrects the error, and then decodes to a target sentence.

3 System Description

This section has discussed OpenNMT, data preparation, pre-processing, system training, and system testing. Figure 3 shows different steps of our system.

3.1 OpenNMT

OpenNMT is an open-source toolkit for Neural Machine Translation, which can also be used for summarization, speech processing etc. It contains an attention mechanism oriented LSTM-based Recurrent Neural Network (RNN) architecture, having pre-processing, training and testing unit. The neural machine (OpenNMT) achieved remarkable performance over human evaluation, rule-based, and statistical machine translation (SMT) systems [6, 15] in large-scale translation tasks such as machine translation, speech recognition etc. Neural networks are appealing since it requires minimal domain knowledge and is conceptually simple [10]. It functions as a black box; when we feed in some inputs from one side, it generates outputs from the other side, and the decision it makes is mainly based on the current information and previously stored output on LSTM (Long Short Term Memory).

3.2 LSTM

LSTMs are the building block of a recurrent neural network comprising a cell, an input gate, an output gate, and a forget gate. Each of these gates is the neuron. The cells are the memory of LSTM responsible for remembering values over arbitrary time intervals. The idea behind LSTM based RNN is how the human brain works; humans don't start their thinking from scratch every second. As we read, we understand each word based on our understanding of the current word and previous word knowledge. When we want to forget everything and want a fresh start, we simply can not forget by deleting everything and start thinking from scratch again; some of our thoughts are persistent though some we forget.

Same way, LSTM works by storing the information in the memory after each iteration and after each epoch. LSTMs (Long Short Term Memory) are capable of learning long-term dependencies. They were introduced by Schmidhuber et al. [5] work tremendously well on a large variety of problems. Figure 1 shows the LSTM used in OpenNMT, where it is used to store data at each iteration and every hidden layer.

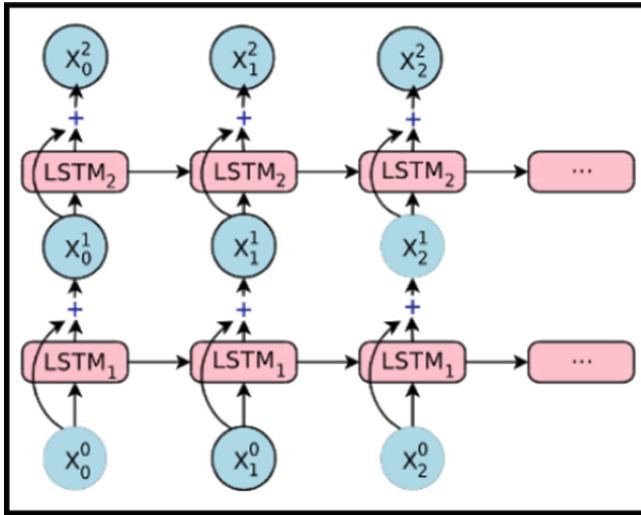


Fig. 1. Block diagram of the LSTM-based system

3.3 Encoder and Decoder of RNN

The encoder and decoder consist of unidirectional RNNs. An encoder in a neural network reads and encodes a source sentence into a fixed-length vector until the end of the sentence is reached and then starts to emit each target word at a time, as Fig. 2 shows. The decoder computes the following hidden states from the previous states based on word embedding, previous target word, and the conditional input derived from the output of the encoder. OpenNMT uses the sequence to sequence LSTM based RNN. While giving inputs as A, B, C and D in the input layer, it generates X, Y and Z as summaries after a number of the hidden layer.

Its attention allows training the neural network by allowing models to learn alignments between different modules. The whole encoder-decoder system is jointly trained to maximize the probability of a correct summary given a source article. In each layer, the encoder reads the input sequence of the source sentence and generates a sequence of states. This source sentence is feed to two unidirectional stacks of LSTM based RNN for the next hidden layer. One stack content is used as it is, and the other is reversed, and these stack layers are concatenated at each linear layer to yield the next layer.

3.4 Rare or Unseen Words

In abstractive text summarization, the critical challenge is to understand the concepts by finding the key entities from the sentence around which the summary revolves. The system prepares the vocabulary of words during the training phase from the training articles. The vocabulary is a word vector having only those

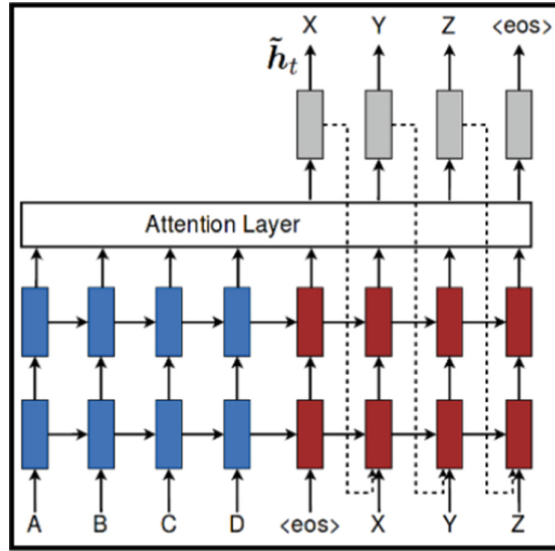


Fig. 2. Neural machine translation - a stacking recurrent architecture for translating a source article A B C D into a target Summary X Y Z. Here, eos marks the end of a sentence.

words of the training article. As a consequence, when LSTM based RNN maps the test article words to the vocabulary, it finds some of the terms are unknown. These new words or out of vocabulary words are called rare or unseen as they are unseen by the model. Hence RNN puts $\langle unk \rangle$ keyword as an unknown token in place of those out of vocabulary words.

3.5 Data Preparation

We performed on DUC 2003, DUC 2004 and Gigaword corpus having news data collected in the raw form. After collecting the raw corpus, we converted it to text format for the convenience of our system. We then performed pre-processing of the raw data to increase the efficiency in terms of space and time by removing noise and other irreverent data. The whole corpus collected was considerably large and required a significant amount of time to train the model. Due to this reason, for faster output generation and evaluation during system training, we use a subset of the collected corpus each time. Careful selection of training and validation datasets is required to avoid over-fitting and underfitting. To avoid overfitting, we selected training and validation data from each domain in the form of article summary pair. The number of instances taken was a ten: one ratio between the training and validation dataset. Similarly, for testing, we extracted the article summary pair. Source article to test the system and the corresponding summary concerning system-generated summary is used further to evaluate the system's accuracy with the help of human evaluator and ROUGE. Input for

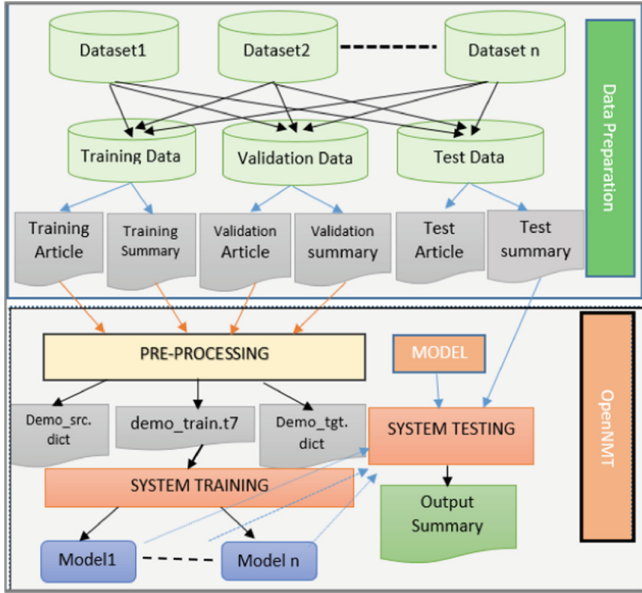


Fig. 3. Block diagram of the system.

pre-processing phase is hence, selected and prepared from the raw news corpus. Figure 3 shows the collection of datasets, and the selection of domain-specific data for training, validation and testing is made for better results.

3.6 Pre-processing

The pre-processing phase tokenizes the datasets based on tab separation and ensures article-summary mapping is present [6]. It eliminates all the unmapped data pairs because, for system accuracy, a mapped parallel article summary is essential. The pre-processing module of OpenNMT accepts the four text files for pre-processing. Namely, trainArticle.txt, trainingSummary.txt, validationArticle.txt and validationSummary.txt files, which we prepared from the raw corpus during the data preparation phase. It generates two human-readable file demo_src.dict, demo_tgt.dict and one Torch file, namely demo.train.t7 file for system training.

3.7 System Training

The output of pre-processing phase of OpenNMT, Torch file (containing all relevant data for training), is used for the training model. Before training, data is shuffled and sorted to ensure instances in the training batch uniformly come from different parts of the corpus. An epoch in training refers to one forward and one backward pass over all the training instances in batches.

Table 1. Details of system parameters used in training

Parameter	Meaning of the parameter
Epoch	One forward pass and one backward pass of all the training examples
Batch size	The number of training examples in one forward/backward pass. The higher the batch size, the more memory space you'll need
Iterations	Number of passes, each pass using [batch size] number of examples. To be clear, one pass = one forward pass + one backward pass (one forward pass and backward pass as one passe)

Some of the essential parameters are listed in Table 2. The system is trained for some fixed number of epochs specified by the end-epoch parameter. An epoch comprises of iterations, and each iteration is of one forward, and one backward pass is performed over a batch size number of training instances. The system has been trained for 13 epochs in the first experimental setup and 16 and 20 epochs in other setups. A validation score, dynamically computed using validation data, helps to check the convergence of training.

3.8 System Testing

For system testing, the source articles are pre-processed to generate tokens and word vectors. The output of each epoch of system training is a model which is capable of generating summaries. System testing with the OpenNMT summarization system uses these trained models to predict summaries for the test articles. The summary generation process makes use of beam search, a heuristic-based optimized version of best-first search, to search the best or optimal summary words. Figure 2 shows the Effectiveness of the search mechanism is exhibited in its ability to facilitate a trade-off between summary generation time and search accuracy, which is ensured by tuning the beam size option of beam search to a relatively small value. A comparison is made between test word vectors and the training system vocabulary to predict the output summary based on semantic similarity. The highest probable words are picked based on the prior knowledge as well as present input from the test article to build the summary. As in our corpus, source sentences are approximate 30 words, and the expected target summary is of a maximum of 8 to 12 words. Besides, the generator uses the 'unk' symbol when it finds rare words. These rare or unseen words are not present in the training models vocabulary. So when mapping is done between the test article to the vocabulary, these words are absent. Hence, the system puts these as an unknown symbol.

4 Experimental Design

We have conducted extensive analysis to better understand our models in terms of learning, the ability to handle different lengths of articles and choices of attentional architectures. This section contains a detailed description of the corpora

and experimental setup used for training and testing the Summarization effectiveness of a Neural Network-based summarization system.

4.1 Corpora Description

In this series of experiments we used open-source DUC² (Document Understanding Conferences) Corpus namely DUC 2003, DUC 2004 and Gigaword³ news corpora, comprises of parallel running article-summary pairs. The DUC corpus of 2004 consists of 624 article-summary pairs, and DUC 2003 consists of 500 article-summary pairs. Annotated English Gigaword was developed by Johns Hopkins Universities Human Language Technology Center of Excellence. Annotated English Gigaword contains nearly ten million documents (over four billion words) is collected from the original English Gigaword, the fifth edition from seven news sources. A subset of news corpus containing 50000, 5000 and 1000 instances is used for training, validation and testing purposes, respectively, randomly selected from the datasets.

4.2 Experimental Setup

We have used the following different experimental setups to train, test and analyze the system's performance from different perspectives.

1. Initially, we trained the system using randomly selected parallel article-summary pairs comprised of 10000 training and 1000 validation instances. Training is done for 13 epochs. These Trained models were tested using 1000 article-summary pairs selected from the same domain of training data.
2. We have re-trained the system using 55000 article-summary pairs, selected from the various corpus and saved the trained model obtained at 16 different epochs after analyzing the system result of the first system. Each of these 16 models has been tested using test corpus, and obtained results have been evaluated using ROUGE evaluation. Such a setup helps us to analyze the change in behaviour of neural machine-based summarization systems with an increase in the number of epochs and training datasets.
3. After successful training of both the model, we created a test dataset of 100 relevant sentences for faster output generation. These sample sentences are tested using 13 epochs of two systems. This helped us to compare both the performance of the system and the output result score using ROUGE are plotted in the graph are shown in Figure 4.
4. Furthermore, we have created different test sets from the original test data, each test set containing 50 articles. The average length of summaries is expected to be 5, 10 and 20, respectively. We selected the best two models from each system based on previous results, and these models are tested using the three test datasets, and prediction results have been evaluated using

² <http://duc.nist.gov/>.

³ <https://catalog ldc.upenn.edu/LDC2002T31>.

ROUGE evaluation. Such a setup helps us to understand the relationship between summary performance and the average length of article summaries in the test dataset. All these summaries are analyzed by comparing with target summary as well as source article.

The results of all these experimental setups have been detailed and analyzed in the result analysis section.

5 Result Analysis

Prediction results of our experiments were self-analyzed and evaluated using ROUGE evaluation [7]. Human evaluator's assessed the quality of the summary with respect to adequacy, length and overall rating. They compared system-generated summary and target summary of test datasets.

Recall-Oriented Understudy for Gisting Evaluation (ROUGE) includes measures to automatically determine the quality of a summary by comparing it to system-generated summaries to target summaries created by humans. The measures count the number of overlapping units such as n-gram, word sequences, and word pairs between the systems generated summary to be evaluated and the ideal summaries created by humans. We used ROUGE 2.0 to evaluate our system, where we learned the ROUGH L, ROUGE 2, and ROUGE 1 score of comparison of summaries in terms of Precision, Recall and F-score.

Precision (P): It is the positive predictive value, i.e. fraction of relevant instances among the retrieved instances. Precision helps us to predict how many words are correct out of all the system generated summary words. In Fig. 5, Precision is TP (True Positive) divided by TP (True Positive) and FP (False Positive), which is the number of words occurring in both systems and target words, i.e. intersection of words between both the summaries divided by the number of words in the system summary.

$$Precision(P) = \frac{\text{number of identical words in both the summary}}{\text{number of words in the system generated summary}}$$

Recall (R): The recall is the fraction of relevant instances that have been retrieved over the total amount of relevant instances. Recall helps us to predict how many words are correctly identified out of all the target summary words. In Fig.5, recall is a fraction of TP (True Positive) and summation of TP (True Positive) and FN (False Negative), which is the number of words occurring in both system and target summaries, i.e.intersection of words between both the summaries divided by the number of words in the ideal summary.

$$Recall(R) = \frac{\text{number of identical words in both the summary}}{\text{number of words in the actual summary}}$$

F-Score: It is a composite measure that combines precision and recall, which depicts how well our system performs. The basic way to compute the F-score is to count the harmonic average of precision and recall:

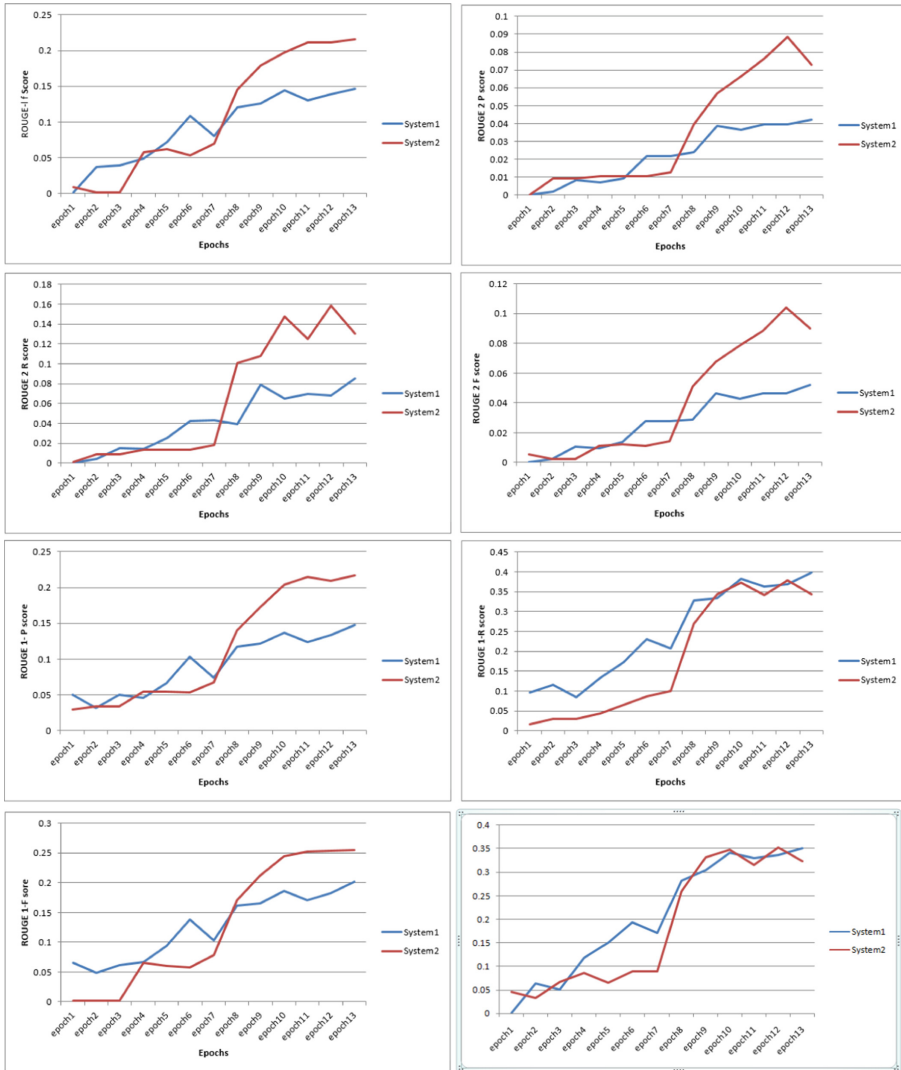


Fig. 4. ROUGE score achieved by system1 and system2 - shown are images of the attention weights learned by various rough score. from the top left ROUGE L-F Score, ROUGE 2-P Score, ROUGE 2-R Score, ROUGE 2-F Score, ROUGE 1-P Score, ROUGE 1-R Score, ROUGE 1-F and ROUGE L-R Score respectively

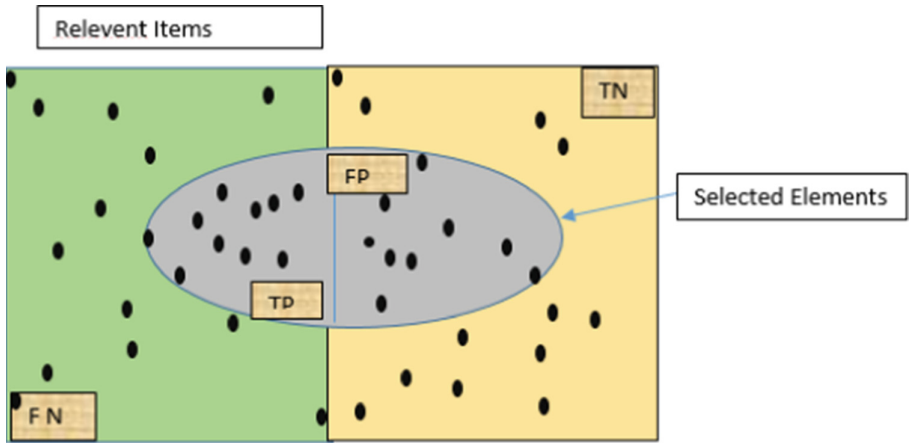


Fig. 5. Data overview: total data can be divided into four group, TP (True Positive), TN (True Negative), FP (False Positive) and FN (False Negative)

$$F - score = 2 * \frac{P * R}{P + R}$$

ROUGE-N, ROUGE-S and ROUGE-L are the granularity of texts being compared between the system summaries and reference summaries. For example, ROUGE-1 refers to the overlap of uni-grams between the system generated summary and target summary. ROUGE-2 refers to the overlap of bi-grams between the system generated summaries and target summaries. ROUGE-L - measures the longest matching sequence of words between both summaries. For example, for this two sentences given below:

System Summary: India vs Pakistan

Target Summary: India Wins Match Against Pakistan

Here in the given instances, System Summary has three words, Target Summary has six words, and the number of common words between the System Summary and Target Summary is 2. ROUGE-1 refers to the overlap of uni-grams between two summaries. So for ROUGE-1 corresponding recall, precision and F-score will be:

$$ROUGE - 1recall = \frac{2}{6} = 0.33$$

$$ROUGE - 1precision = \frac{2}{3} = 0.66$$

$$ROUGE - 1F - score = 2 * \frac{(0.33 * 0.66)}{0.33 + 0.66} = 0.44$$

If the target summary is larger than the system summary, precision may give a good score, and if system generated summary is larger than recall will give a good score. Finally, the score generated at each testing is the average of

individual scores. So, we get nine scores at every output P, R, and F of ROUGE-L, ROUGE-1 and ROUGE-2 each. Figure 4 shows a comparison of ROUGE L-F Score, ROUGE 2-P Score, ROUGE 2-R Score, ROUGE 2-F Score, ROUGE 1-P Score, ROUGE 1-R Score, ROUGE 1-F and ROUGE L-R Score between two best systems 13 epochs.

Table 2 shows the ROUGE score achieved by two systems. The ROUGE-1 score of 39.89 is attained at epoch 13 of System 1, and the ROUGE-1 score of 37.04 is achieved in the 14th epoch of System 2. The ROUGE score curve converges the eleventh epoch, and we also analyzed that for short sentences, we got better results of 42.03 by system 2.

After analyzing all the epochs output with 100 test data, we Further have created different test sets from the original test data, each test set containing 50 sentences. The average length of sentences in the three test datasets is 8, 15 and 20 words, respectively. Table 2 depicts some of the analyzed results. We came to the conclusion that small sentences give a better summary in terms of ROUGE score.

Table 2. Best experimental results of our system

System used	Test data	Best ROUGE-1 score
System 1 13 epoch	100 sentence	39.89
System 2 14 epoch	100 sentence	37.04
System 2 14 epoch	50 sentence	39.89
System 2 14 epoch	50 short sentence	42.03

We have also analyzed some summaries. These are presented in Table 3 shows some of the systems generated summaries of both the system generated from the same source sentences. After analyzing the results, we came to the conclusion that though summaries are not exactly the same as the target summary or the same in all the systems, summaries generated by the best models (epoch) are semantically correct to a large extent.

Table 3. Sample summary predictions

Prediction by system 1	Prediction by system 2
Palestinian minister heads for	Palestinian fm says he will not stand down
French prime minister arrives in rome	French prime minister arrives in sarajevo
Malaysia’s lavrov calls for international cooperation	Malaysia’s defense minister calls for reconciliation
Nigerian central banks step up more than ## million dollars	Nigerian central african leaders to discuss ceasefire
Suicide bomber kills ## in turkey	## killed in iraq chopper crash

6 Conclusion and Future Works

In this paper, we applied the attention-based sequence to sequence recurrent neural networks for abstractive text summarization. The attentional recurrent neural network allows training the neural network by allowing the models to learn alignments between different modulus using LSTMs (Long Short Term Memory). It functions as a black box; when we feed in some inputs from one side, it generates some outputs from the other side, and the decision it makes is mostly based on the current inputs and previously-stored output based on LSTMs, which are a special kind of RNN.

RNN based summarization system relies heavily on the size of the training corpus, which motivated us to use a significant amount of training corpus. The Effectiveness of summarization is largely determined by the attention mechanism and the score function used for computing attention of each hidden state and error correction. A practised and careful selection of values for system parameters such as the number of epochs, batches, hidden layers etc., can also significantly improve the summarization quality. We have trained, tested and analyzed the proposed systems for summarization using various news summarization datasets. Predicted summaries have been evaluated using ROUGE evaluation. The human evaluator analyzed the quality of summarization in terms of its adequacy, quality and redundancy and found that after a certain number of epochs, the trained models are able to produce semantically correct summaries.

In the next work, we will extend our efforts on this corpus and build more robust models compared to our baseline system for a more accurate summary generation, such that it can also handle unknown words. The model can be further extended to multi-lingual and multi-document automatic summarization tasks.

References

1. Baxendale, P.B.: Machine-made index for technical literature-an experiment. *IBM J. Res. Dev.* **2**(4), 354–361 (1958)
2. Cho, K., et al.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint [arXiv:1406.1078](https://arxiv.org/abs/1406.1078) (2014)
3. Clarke, J., Lapata, M.: Discourse constraints for document compression. *Comput. Linguist.* **36**(3), 411–441 (2010)
4. Edmundson, H.P.: New methods in automatic extracting. *J. ACM (JACM)* **16**(2), 264–285 (1969)
5. Gers, F.A., Schmidhuber, J., Cummins, F.: Learning to forget: continual prediction with LSTM. *Neural Comput.* **12**(10), 2451–2471 (2000)
6. Klein, G., Kim, Y., Deng, Y., Senellart, J., Rush, A.M.: OpenNMT: open-source toolkit for neural machine translation. arXiv preprint [arXiv:1701.02810](https://arxiv.org/abs/1701.02810) (2017)
7. Lin, C.Y.: Improving summarization performance by sentence compression-a pilot study. In: *Proceedings of the Sixth International Workshop on Information Retrieval with Asian Languages*, pp. 1–8 (2003)
8. Lin, C.Y.: Rouge: a package for automatic evaluation of summaries. In: *Text Summarization Branches Out*, pp. 74–81 (2004)

9. Luhn, H.P.: The automatic creation of literature abstracts. *IBM J. Res. Dev.* **2**(2), 159–165 (1958)
10. Luong, M.T., Pham, H., Manning, C.D.: Effective approaches to attention-based neural machine translation. arXiv preprint [arXiv:1508.04025](https://arxiv.org/abs/1508.04025) (2015)
11. Rush, A.M., Chopra, S., Weston, J.: A neural attention model for abstractive sentence summarization. arXiv preprint [arXiv:1509.00685](https://arxiv.org/abs/1509.00685) (2015)
12. Shardan, R., Kulkarni, U.: Implementation and evaluation of evolutionary connectionist approaches to automated text summarization (2010)
13. Sutskever Ilya, V.O., V, L.Q.: Sequence to Sequence Learning with Neural Networks, pp. 3104–3112. Curran Associates, Inc. (2014)
14. Wang, S., Zhao, X., Li, B., Ge, B., Tang, D.: Integrating extractive and abstractive models for long text summarization. In: 2017 IEEE International Congress on Big Data (BigData Congress), pp. 305–312. IEEE (2017)
15. Wu, Y., et al.: Google’s neural machine translation system: bridging the gap between human and machine translation. arXiv preprint [arXiv:1609.08144](https://arxiv.org/abs/1609.08144) (2016)

Text Mining



Using Reviewer Information to Improve Performance of Low-Quality Review Detection

Qingliang Miao^(✉), Changjian Hu, and Feiyu Xu

Lenovo Research, No. 6 Shangdi West Road, Haidian District, Beijing, China
{miaoql1, hucj1, fxu}@lenovo.com

Abstract. The drastic increase of user-generated contents has exhibited a rich source for mining opinions. Unfortunately, the quality of user-generated content varies significantly from excellent to meaningless, which by general estimation, causes a great deal of difficulty in mining-related applications. In the field of low-quality review detection, many previous approaches have individually detected low-quality reviews by using the intrinsic features of the review. However, no systematic study measuring the significance of reviewer information for detecting low-quality reviews has been previously done. In this paper, the importance of reviewer information when predicting review quality is studied and how to exploit it to build low-quality review detection models is determined. The experimental results on two different domains show that reviewer information does matter when modeling and predicting the quality of reviews. It is also shown that significant performance improvements can be achieved if the reviewer information is integrated with the intrinsic features of the reviews. These findings are of the essence in solving the low-quality review detection problem and in developing review-based opinion mining applications.

Keywords: Mining opinions · Reviewer information · Quality of reviews

1 Introduction

With the dramatic development of Web 2.0, user-generated contents have become increasingly prevalent on the web. Popular user-generated contents include reviews on e-commerce websites, blogs, and web forums. However, due to the absence of editorial and quality control, user-generated contents vary greatly in quality, which in general estimation, causes problems in mining applications, such as opinion extraction [1, 2], sentiment classification [3, 4], and opinion summary [6]. In the opinion retrieval domain, the quality of a review can also be incorporated into retrieval models in the form of a prior probability [10].

Product reviews are widely used to mine customers' opinions on products. So review-based opinion mining can be more effective if low-quality reviews are preliminarily filtered. According to [6], low-quality reviews are reviews that have little or incorrect description of a product, have little or no comments on some aspects of the product, and do not provide convincing opinions with sufficient supporting evidence. Figures 1 (a) and

(b), respectively, show review and reviewer information from a high-quality review. The high-quality review shown in Fig. 1 (a) describes several aspects of the product, such as appearance, image quality, response, and battery, and provides convincing opinions with sufficient supporting evidence. Figure 1 (b) shows that the reviewer has published four reviews: one is about video games, and the other three are on digital cameras. Moreover, it can be seen that the helpful votes and total votes for the reviewer is very high, (17/17, 403/409, 11/12, 25/25). Based on this reviewer information, it may be concluded that the reviewer is likely to publish high-quality reviews, especially in the digital camera domain.

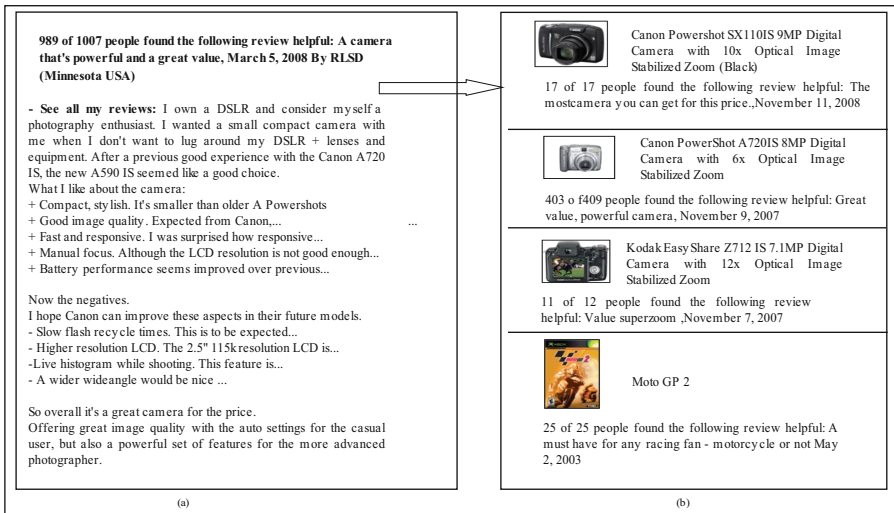


Fig. 1. A segment of review and reviewer information of a high-quality review.

In Fig. 2 (a), the reviewer gives little useful information about the product, but complains of an unsatisfactory experience with the camera. It can be seen, in Fig. 2 (b), that the reviews did not receive much peer-to-peer voting, (4/18, 2/22, 4/14). Intuitively, it may be expected that the reviewer is unlikely to publish high-quality reviews.

At present, most e-commerce websites allow users to evaluate the quality of the reviews by assigning helpful votes to them. For example, Amazon.com provides review readers with a mechanism for judging whether a review is helpful or not. The mechanism accumulates helpful votes from a particular review, and the number of helpful votes a review receives indicates its actual effectiveness. For convenience, in this paper, this mechanism is called peer-to-peer voting, which is a good way to assess the quality of reviews. However, the mechanism is not effective in the following cases [5]: (1) newly-written reviews cannot be evaluated immediately, because they need to accumulate peer to peer votes; and (2) low-traffic reviews and reviews with few helpful votes similarly cannot be evaluated. Therefore, it is vital to have the ability to detect low-quality reviews automatically, especially newly-written and low-traffic reviews.

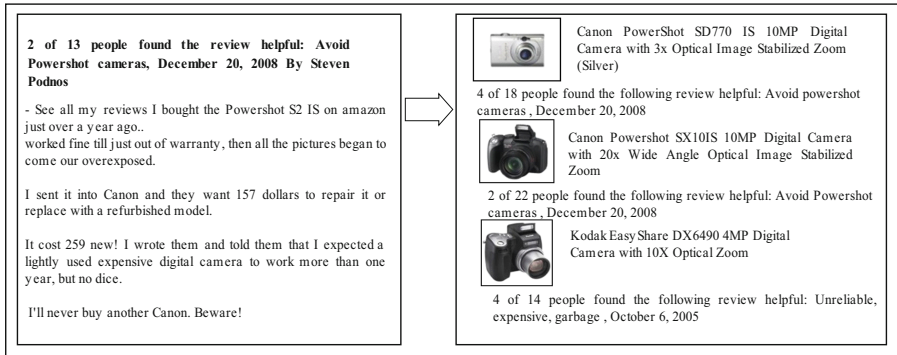


Fig. 2. A segment of review and reviewer information of a low-quality review

The task of detecting low-quality reviews is presently treated as a binary classification problem [6] or a regression problem [5, 7]. Several methods applying only the intrinsic features to assess review helpfulness have been reported. These approaches are based either on lexical or syntactic features, along with semantic features. Information from high-quality answer findings in the question-answering community has proven to be very helpful in estimating the quality of the answers [8, 9]. In [16], Zhang presented a preliminary analysis of whether author knowledge was a powerful usefulness predictor and drew the conclusion that authorship did seem to be a powerful usefulness predictor. However, how to acquire and exploit reviewer information should be further investigated. Inspired by [8, 9, 16], it is hypothesized that the quality of reviewer information could improve the performance of a low-quality review detection model. Therefore, one of the focuses of this study is to further demonstrate whether reviewer features have potential for predicting review quality.

The research questions described above can be summarized as follows:

1. Does the quality of the reviewer information matter for building better models to detect low-quality reviews?
2. Which reviewer features are most predictive in low-quality review detection models?
3. How do we acquire and exploit reviewer information to improve the performance of the low-quality reviews detection model?

These questions are answered by conducting an empirical study on two real world datasets, under different experimental conditions. To answer the first question, two kinds of low-quality review detection models were built, one included reviewer information and the other did not, and their detection performances were compared. To answer the second question, various reviewer features were selected as a baseline and then other reviewer features were added into the low-quality review detection models, and then their performance in detecting low-quality reviews was compared. For the third question, the derivation of the reviewer information is explained and the reviewer information is translated into reviewer features in low-quality review detection models.

This paper makes the following contributions to the study of reviewer information in low-quality review detection. First, it is systematically demonstrated that the

quality of reviewer information indeed matters when detecting low-quality reviews in some domains. Second, the reviewer features that are most predictive are discovered, using our low-quality review detection models. Third, the acquisition and exploitation of reviewer information to improve the performance of low-quality review detection models is explained. Fourth, based on empirical experiment results on electronic products and book domains, it is determined that reviewer information is more effective in the electronic product domain than in the book domain.

The rest of the paper is organized as follows: Sect. 2 contains the literature review. Section 3 presents our approach for the detection of low-quality reviews. In Sect. 4, the study's evaluation criterion is introduced. In Sect. 5, the proposed research questions are empirically demonstrated. Section 6 summarizes the work in this paper and calls attention to future work.

2 Literature Review

Interest in sentiment analysis has recently increased as part of a larger research effort in affective computing [17]. Many approaches on sentiment analysis and feature level-based opinion mining have been proposed.

2.1 Sentiment Analysis

In the field of sentiment analysis, P. Turney [3] proposed a corpus-based approach, PMI-IR, to determine semantic orientation. Theresa Wilson et al. [18] presented the first experimental results classifying the strength of opinions. B. Pang et al. [4] adopted standard machine learning techniques to determine whether a review is positive or negative. Moreover, S. Kim et al. [19] proposed a system to determine the Sentiment of Opinions.

2.2 Feature Level-Based Opinion Mining

There are some approaches for mining product opinion at product feature levels, based on product reviews, and they are usually classified as unsupervised- and supervised-based methods. Representative works of the unsupervised-based method include [2, 20, 21]. In [20] and [21], M. Hu and B. Liu's work is performed in three steps: (1) mining the product features and opinions, (2) identifying the opinion orientation, and (3) summarizing the mining results. Popescu et al. [2] proposed a web-based feature extraction method. In their method, each noun phrase is given a pointwise mutual information score between the phrase and part discriminators associated with the product class. The score is computed by the "KnowItAll" system. Qi Su et al. [23] mainly studied the problem of extracting implicit features from customer reviews; they proposed a feature-based pointwise mutual information algorithm. Carenini et al. [22] proposed a more sophisticated method based on several similarity measures. Their system merges each discovered feature to a feature node in the user-defined taxonomy. The similarity measures are defined based on string similarity, while synonyms and other distances are measured using WordNet. Bin Shi and Kuiyu Chang [24] proposed an "opinion first, feature second" approach. They manually built a hierarchical product feature concept model using product domain knowledge, and

extracted product features based on the concept model. Ronen Feldman, Moshe Fresko, et al. [25] presented a study in extracting comparison information. Representative works describing supervised-based methods include [26] and [27]. Rayid Ghani, Katharina Probst, et al. [26] viewed the product features extraction problem as a classification problem, using single-view and multi-view semi-supervised learning algorithms. Bo Wang and Houfeng Wang [27] considered the fact that product properties and opinion words usually co-occur with high frequency in product review articles and proposed to bootstrap both of them using cross-training.

2.3 Quality Assessment of User-Generated Contents

When mining opinions from reviews or other user-generated contents, it is important to consider whether or not individual reviews are helpful or useful [17]. In the past few years, there has been an increasing interest in automatically assessing the quality of user-generated contents, including product reviews on e-commerce websites, weblogs, question-answer communities, and web forums.

In the field of assessing review helpfulness and detecting low-quality reviews, a representative work is offered by Kim et al. [5], which considered the task as a ranking problem and solved it with regression models. In their experiments, they adopted an SVM regression model and used structural features, lexical features, syntactic features, semantic features, and meta-data features in the process of regression model training. The peer-to-peer voting information, which was derived from Amazon.com, was used as ground-truth. Based on their experimental results, they found that the most useful features included the length of the review (structural feature), the unigrams of the review (lexical feature), and the product rating of the review (meta-data feature). Zhang and Varadarajan [7] proposed a framework that integrated polarity and the utility of the reviews. Within this framework, they also used regression models to predict the utility of the reviews. More specifically, they adopted simple linear regression and e-support vector regression to rank reviews according to utility. In their experiments they found that shallow syntactic features, such as proper nouns and numbers of modal verbs, account for most predicting power of the regression model. Zhang [16] defined a new task in text sentiment analysis, which adds usefulness scoring to opinion extraction to improve product review ranking services and helps shoppers and vendors leverage information from multiple sources. Ghose and Ipeirotis [11] proposed a review ranking mechanism that combines econometric analysis with text mining techniques, and they found that reviews which include a mixture of subjective and objective elements are considered more helpful by users. In addition, they observed that for feature-based goods, such as electronics, users prefer reviews to contain mainly objective information with a small amount of subjective information. However, for experience goods, such as movies, users prefer personalized, highly-sentimental opinions.

There are some other studies that treated the low-quality review detection problem as a binary classification problem. Liu's work [6] may be the most representative research in this area. In their work, they defined a standard specification to measure the quality of product reviews and proposed several intrinsic review features to train a model. They found that sentence level features, word level features, and product characteristic level features were most effective in their experiments. More importantly, they argued that

three types of biases, including imbalance vote bias, winner circle bias, and early bird bias, exist in peer-to-peer voting evaluation standard [5, 7]. Therefore, they hired four annotators to label the reviews manually. In addition, they applied the low-quality review detection approach to enhance opinion summarization and yielded better performance. In other words, the importance of low-quality review detection was validated in their work. The approach used in the present study is different from [5–7, 11]. First, our focus is to further demonstrate whether reviewer information has potential for detecting low-quality reviews. In addition, the reviewer features that are most effective in a low-quality review detection model are discovered.

Some approaches for finding high-quality answers in question-answer communities were also proposed. Agichtein et al. [8] introduced a general classification framework for combining the evidence from different sources of information and investigated methods for exploiting intrinsic content quality and community feedback to automatically identify high quality content. Jeon et al. [9] presented a framework to use non-textual features to predict the quality of documents. To the best of our knowledge, no systematic study measuring whether reviewer information matters for building better models predicting review quality has been previously conducted, which is the focus of this paper. Weimer and Gurevych [13] studied the problem of predicting the quality of web forum posts, and they built a system which learns from human ratings by applying SVM classification. Surface, lexical, syntactic, forum specific, and similarity features were used in the learning process. They tested the model on three datasets and found that surface and forum-specific features are more useful.

3 The Low-Quality Review Detection Approach

Review quality evaluation is an interesting problem, which has many potential applications. For example, it can be used as a pre-processing procedure for review ranking algorithms. In essence, the approach of this study is to exploit features that are intuitively correlated with the quality of user-generated contents, and then train a model to mine the relationship between them. Based on the mined knowledge, the quality of user-generated contents can be evaluated.

3.1 Problem Definition

As previously discussed, low-quality reviews are reviews that have little or incorrect description of a product, have little or no comments on some aspects of the product, and do not provide convincing opinions with sufficient supporting evidence [6]. In other words, low-quality reviews do not provide enough useful information to users. The core of this research includes two main issues: features learning and model selection. The first issue concerns the features that should be selected to model the quality of the reviews, and the second one concerns the learning algorithms that are effective to model the quality of the reviews. In this paper, it is assumed that there are two kinds of reviews in the review space: high-quality reviews and low-quality reviews. Under this assumption, low-quality review detection is treated as a binary classification problem. Formally, given a training data set of high-quality reviews and low-quality reviews,

$T = \{f_i, Y_j\}, i = 1 \dots n; j = 1, 2$, statistical machine learning approaches are adopted to learn classification models that can maximize the accuracy in the classification of Y_i given $f_i, i = 1 \dots n$ where $f_i, i = 1 \dots n$ represents learning features and $Y_j, j = 1, 2$ stands for high-quality and low-quality, respectively. When a new review comes, the classification model automatically assesses high-quality or low-quality to the review.

3.2 The Low-Quality Review Detection Model

As previously discussed, the core of the low-quality review detection model is how to identify the features and how to learn the detection models. For the first issue, previous studies have proved that reviewing intrinsic information is important to model review quality; however whether reviewer information is helpful to model review quality is unknown. Product reviews often involve personal experience, knowledge, and interests; therefore, both the intrinsic information in the review and the reviewer information are taken into consideration. For the second issue, three classification algorithms (Adaboost, C4.5, and SVM) are adopted to learn the low-quality review detection models. In particular, reviews and reviewers' information was collected from Amazon.com, and then both review and reviewer features were extracted as learning features. After features extraction, reviews are labelled as high-quality class or low-quality class. Classification algorithms are then adopted to learn the detection models. Finally, the learned detection models are evaluated using a test dataset.

3.3 The Learning Features

Many previous studies have detected low-quality reviews by using intrinsic review features. One of the focuses in this paper is to further demonstrate whether reviewer information matters for building better models to detect low-quality reviews. If the reviewer information is effective in modeling review quality, which reviewer information is more effective? In the approach here proposed, both review and reviewer features are taken into consideration. User-generated contents are created by millions of end-users; therefore, the quality of the contents is closely correlated with the end-users. "Good" reviewers write "good" reviews. Reviewers' personal experience, knowledge, interests, and reputation are closely related to review quality; therefore, reviewer information can be useful in constructing and optimizing review quality models. In [5–7], researchers have reported which are the effective review features for detecting low-quality reviews, therefore, these effective review features are adopted as a baseline.

Review features

Three categories of review features are chosen, including surface features, structure features, and shallow syntactic features.

Surface and Structure Features:

F1: The total number of tokens in a syntactic analysis of a review [5].

F2: The number of sentences in a review [6].

F3: The average length of sentences [6].

F4: The number of sentences with product features [6].

F5: The number of products in a review [6].

F6: The number of brand names in a review [6].

F7: The number of product features in a review [6].

F8: The total frequency of product features in a review [6].

F9: The average frequency of product features in a review [6].

F10: The number of paragraphs in a review [6].

F11: The average length of paragraphs in a review [6].

Shallow Syntactic Features:

F12: Proper nouns: reference to existing, maybe technical concepts [7].

F13: Modal verbs: reflection of certainty, confidence, mood, etc., which are all instances of modality [7].

F14: Interjections: signals of emotion [7].

F15: Comparative and superlative adjectives: indicators of comparison [7].

F16: Comparative and superlative adverbs: also indicators of comparison [7].

F17: wh-determiners, wh-pronouns, possessive wh-pronouns, wh-adverbs: wh-words signify either questions or other interesting linguistic constructs, such as relative clauses [7].

Features F1, F2, F3, F5, F6, F10, and F11 can be easily derived from the review itself. Features F4, F7, F8, and F9, cannot be directly obtained. An integration strategy [12] is adopted to mine the product features, and then they are computed. For features F12 to F17, Stanford POS tagger is used; the POS taggers of F12 to F17 are {NNP}, {MD}, {UH}, {JJR}, {JJS}, and {WDT, WP, WP\$, WRB}.

Review features

In order to exploit reviewer information, it has to be translated into reviewer features. Eight reviewer features are introduced in this paper.

F18: The total number of reviews the reviewer has written.

Our hypothesis is that if a reviewer has published many reviews, his reviews are likely high-quality.

F19: The sum of helpful votes the reviewer has received.

Our hypothesis is that if a reviewer has received many helpful votes from other people, then his reviews are likely high-quality.

F20: The total votes the reviewer has received.

F21: The average total votes the reviewer has received.

F22: The average helpful votes the reviewer has received.

Note that since the helpful votes and total votes information will be used in Sect. 4 to label the reviews as high-quality or low-quality, features F19 to F22 do not contain the helpful votes and total votes information of the review that is to be classified.

F23: The reviewer's domain authority.

Our hypothesis is that if a reviewer has published many reviews in a domain, he or she is authoritative in that domain. Under certain conditions, a reviewer may write many reviews that belong to different domains. In this case, different weights are assigned to different domains. For example, a reviewer writes electronic products reviews and he/she also writes book and movie reviews. When the review in the electronic product domain is classified, more weight will be assigned to it.

F24: The rating score the reviewer assesses in a review.

The rating score is from one star to five stars.

F25: The Kullback-Leibler distance between the rating score and the average rating score given by all reviewers.

In order to derive the features F18 to F25, reviewers' information is collected, including total reviews, ranting, helpful votes, and total votes from Amazon.com.

4 Evaluation Criterion

As previously discussed, there are two kinds of evaluation methods: peer-to-peer voting evaluation [5, 7, 16], and manual annotation evaluation [6]. In [5], Kim et al. made use of peer-to-peer voting information to evaluate the quality of reviews and defined a review helpfulness function as:

$$h(r \in R) = \frac{\text{rating}_+(r)}{\text{rating}_+(r) + \text{rating}_-(r)}$$

where $\text{rating}_+(r)$ is the number of people that will find a review helpful and $\text{rating}_-(r)$ is the number of people that will not find the review helpful. In [6], Liu et al. argued that there are three kinds of biases in peer-to-peer voting evaluation: imbalance vote bias, winner circle bias, and early bird bias. Therefore, manual annotation evaluation was adopted in this study. The definitions of high-quality reviews and low-quality reviews are as follows [6]:

High-quality review: A review should contain a complete or relatively complete comment on a product and features of the product. Moreover, it should provide convincing opinions with sufficient supporting evidence. It provides practical information to the user.

Low-quality review: A review provides little useful information or gives misleading information. It does not help the user in making a decision.

5 Experiment Setup

In this section, datasets are first introduced, and then the experimental results in answer to the research questions are provided. In our experiment, three popular learning models were adopted: Adaboost, C4.5, and SVM.

5.1 Dataset

Two product types were chosen, namely electronic products and books. We selected 1,756 electronic products and 2,035 books as seeds. Then, the ASIN of each product was transferred to Amazon Web Services API and 72,072 electronic product reviews and 92,212 book reviews were obtained. Using Amazon Web Services API, 41,722 and 44,588 pieces of reviewers' information were obtained in the electronic products and book domains, respectively. Figures 3 and 4 show the distribution characteristics of the reviews and reviewers. Figure 3 shows that a large number of products get very few reviews and a small number of products get a large number of reviews. Therefore, the products with less than 50 reviews were dropped, resulting in 414 electronic products

with 41,360 reviews and 477 books with 44,653 reviews. Figure 4 indicates that a large number of reviewers write only a few reviews, and a few reviewers write a large number of reviews. In order to test the effectiveness of reviewer features, the reviewers who only wrote two reviews were removed. Finally, a dataset of 29,645 electronic products reviews and 29,776 book reviews was obtained. Six people were employed to annotate the data, according to the criterion proposed in Sect. 4. The statistics of the dataset are shown in Table 1.

Table 1. The statistics of the datasets.

Domain	#Product	#Reviews	#High Quality	#Low Quality
Electronic Products	414	29645	11877	17768
Books	477	29776	11604	18172

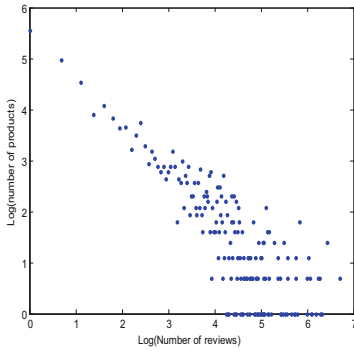


Fig. 3. The log-log plot of the number of reviews to the number of products

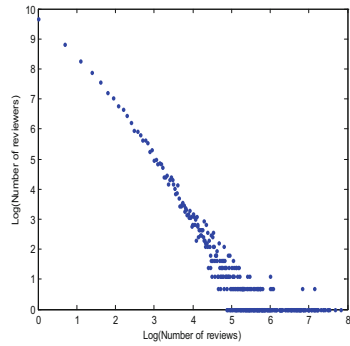


Fig. 4. The log-log plot of the number of reviews to the number of reviewers

5.2 Performance Measures

Six performance measures were used in these experiments, including Accuracy, Precision, Recall, F-Measure, RUC, and ROC curves, to evaluate the effects of the reviewer information.

$$accuracy = \frac{TP + TN}{TP + FP + TN + FN} \quad precision = \frac{TP}{TP + FP} \quad recall = \frac{TP}{TP + FN} \quad F - measure = \frac{2precision * recall}{precision + recall}$$

TN is the number of negative examples correctly classified (True Negatives), FP is the number of negative examples incorrectly classified as positive (False Positives), FN is the number of positive examples incorrectly classified as negative (False Negatives) and TP is the number of positive examples correctly classified (True Positives). ROC curves can be thought of as representing the family of best decision boundaries for relative costs of TP and FP [14]. On an ROC curve the X-axis represents False Positive Rate = $FP / (TN + FP)$ and the Y-axis represents True Positive Rate = $TP / (TP + FN)$. The AUC (area under the ROC curve) is a useful metric for classifier performance, as it is independent of the decision criterion selected and prior probabilities [14].

5.3 Does Reviewer Information Matter?

The aim of this study is to experimentally demonstrate whether reviewer information matters when predicting review quality. Therefore, two detection models were constructed and their predictive performances were compared. One model included reviewer information and the other model did not.

Table 2. The experiment results for the electronic product domain using Adaboost

Features	Accuracy	Precision	Recall	F-Measure	AUC
Review-features	0.734	0.755	0.822	0.787	0.80
Reviewer-features	0.782	0.801	0.846	0.823	0.864
All-features	0.838	0.861	0.870	0.866	0.913

Table 3. The experiment results for the electronic product domain using C4.5

Features	Accuracy	Precision	Recall	F-Measure	AUC
Review-features	0.732	0.743	0.845	0.791	0.768
Reviewer-features	0.765	0.791	0.825	0.808	0.831
All-features	0.812	0.842	0.847	0.844	0.858

Table 4. The experiment results for the electronic product domain using SVM

Features	Accuracy	Precision	Recall	F-Measure	AUC
Review-features	0.766	0.774	0.862	0.815	0.742
Reviewer-features	0.790	0.809	0.850	0.829	0.775
All-features	0.838	0.916	0.803	0.856	0.846

Table 2, 3 and 4 show the experiment results for the electronic products domain using Adaboost, C4.5, and SVM.

Table 2, illustrates that reviewer features are more effective than review features on all measures. Comparing review features and reviewer features, the improvements for Accuracy, F-Measure, and AUC are 4.8%, 3.6%, and 6.4%, respectively. When training the learning model using both review features and reviewer features, the performance is more improved than when using review features only. Accuracy, F-Measure, and AUC are improved by 10.4%, 7.9%, and 11.3%, respectively, when using review and reviewer features together. From Table 3, it can also be seen that reviewer features perform better than review features. Comparing review features and reviewer features, the improvements of Accuracy, F-Measure, and AUC are 3.3%, 1.7%, and 6.3%, respectively; when using reviewer features and review features together, Accuracy, F-Measure, and AUC

improved by 8.0%, 5.3%, and 9.0%, respectively. In Table 4, comparing review features and reviewer features, the improvements of Accuracy, F-Measure, and AUC are 2.4%, 1.4%, and 3.3%, respectively; when using reviewer features and review features together, Accuracy, F-Measure, and AUC improved by 7.2%, 4.1%, and 10.4%, respectively.

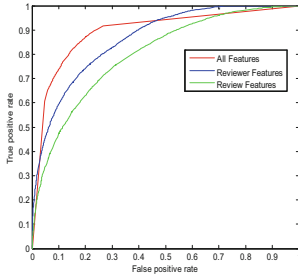


Fig. 5. ROC curves on different features of the electronic product domain using Adaboost

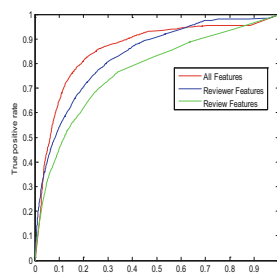


Fig. 6. ROC curves on different features of the electronic product domain using C4.5

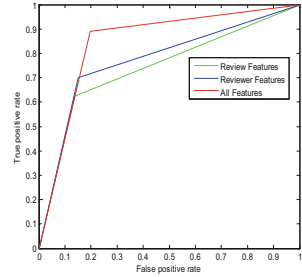


Fig. 7. ROC curves on different features of the electronic product domain using SVM

Figure 5 shows the ROC curves on different features of the electronic product domain using Adaboost. The ROC curve of reviewer features is above the ROC curve of review features, therefore, reviewer features perform better than review features. Used together, reviewer features and review features provide the best detection performance. Figure 6 shows the ROC curves on different features of the electronic product domain using C4.5. From Fig. 6, the same conclusion can be drawn. Figure 7 shows the ROC curves on different features of the electronic product domain using SVM. Again, the same conclusion can be drawn. From the above analysis, it can clearly be seen that reviewer information indeed matters when predicting review quality in the electronic product domain.

In order to further demonstrate whether reviewer features can improve low-quality review detection performance in other domains, a comparative trial was conducted in the book domain.

Table 5. The experiment results for the book domain using Adaboost

Features	Accuracy	Precision	Recall	F-Measure	AUC
Review-features	0.657	0.692	0.792	0.738	0.682
Reviewer-features	0.720	0.754	0.804	0.778	0.797
All-features	0.721	0.763	0.789	0.775	0.798

Table 6. The experiment results for the book domain using C4.5

Features	Accuracy	Precision	Recall	F-Measure	AUC
Review-features	0.658	0.694	0.786	0.737	0.632
Reviewer-features	0.733	0.751	0.840	0.793	0.787
All-features	0.747	0.770	0.835	0.801	0.810

Table 7. The experiment results for the book domain using SVM

Features	Accuracy	Precision	Recall	F-Measure	AUC
Review-features	0.639	0.673	0.795	0.729	0.595
Reviewer-features	0.698	0.705	0.868	0.778	0.65
All-features	0.636	0.632	0.965	0.764	0.543

Table 5, 6 and 7 show the experiment results for the book domain using Adaboost, C4.5, SVM. Table 5 shows that reviewer features perform better than review features. Comparing review features and reviewer features, the improvements of Accuracy, F-Measure, and AUC are 6.3%, 4.0%, and 11.5%, respectively. When training the learning model using both review features and reviewer features together, Accuracy, F-Measure, and AUC improved by 6.4%, 3.7%, and 11.6%, respectively. Accuracy and F-Measure improved less than AUC. From Table 6, it can be seen that reviewer features perform better than review features on all measures, and the improvements of Accuracy, F-Measure, and AUC are 7.5%, 5.6%, and 15.5%, respectively. When using reviewer features and review features together, Accuracy, F-Measure, and AUC improved by 8.9%, 6.4%, and 17.8%, respectively. Table 7 shows that reviewer features perform better than review features on all measures, and the improvements of Accuracy, F-Measure, and AUC are 5.9%, 4.9%, and 5.5%, respectively. Surprisingly, when using reviewer and review features together, only recall improved by 17%.

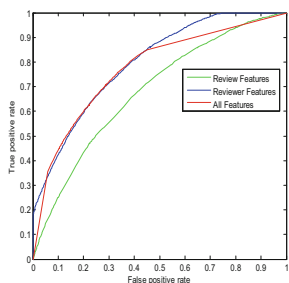
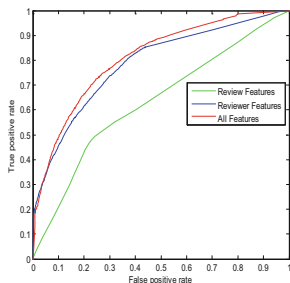
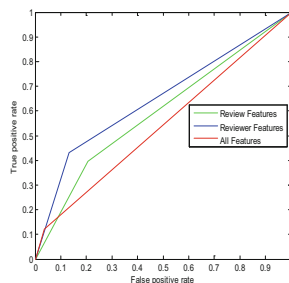
**Fig. 8.** ROC curves on different features of the book domain using Adaboost**Fig. 9.** ROC curves on different features of the book domain using C4.5**Fig. 10.** ROC curves on different features of the book domain using SVM

Figure 8 shows the ROC curves on the different features of the book domain using Adaboost. Figure 8 shows that the ROC curve of reviewer features is above the ROC curve of review features, therefore, reviewer features perform better than review features. Reviewer and review features used together result in the best detection performance. Figure 9 shows the ROC curves on different features of the book domain using C4.5. From Fig. 9, the same conclusion can be drawn: reviewer features improve the performance of the detection model. Figure 10 shows the ROC curves on different features of the book domain using SVM, and indicates that reviewer features perform best.

5.4 Which Reviewer Features are Most Predictive?

In this experiment, reviewer features F19 and F20 were used as a baseline, and other reviewer features were incrementally added to the training process. The experiment results of the electronic product domain and the book domain, using Adaboost, C4.5, and SVM are shown in Table 8 to Table 13.

Table 8. The results for reviewer features of the electronic product domain using Adaboost

Reviewer Features	Accuracy	Precision	Recall	F-Measure	AUC
F19, 20	0.718	0.766	0.763	0.765	0.784
F19, 20, 21, 22	0.748	0.787	0.795	0.791	0.809
F19, 20, 21, 22, 18	0.760	0.787	0.821	0.804	0.845
F19, 20, 21, 22, 18, 23	0.767	0.789	0.835	0.811	0.851
F19, 20, 21, 22, 18, 23, 24	0.780	0.802	0.841	0.821	0.861
F19, 20, 21, 22, 18, 23, 24, 25	0.782	0.801	0.846	0.823	0.864

Table 9. The results for reviewer features of the electronic product domain using C4.5

Reviewer Features	Accuracy	Precision	Recall	F-Measure	AUC
F19, 20	0.719	0.782	0.736	0.759	0.772
F19, 20, 21, 22	0.749	0.791	0.79	0.79	0.795
F19, 20, 21, 22, 18	0.755	0.782	0.82	0.801	0.819
F19, 20, 21, 22, 18, 23	0.759	0.788	0.818	0.802	0.825
F19, 20, 21, 22, 18, 23, 24	0.764	0.789	0.826	0.807	0.832
F19, 20, 21, 22, 18, 23, 24, 25	0.765	0.791	0.825	0.808	0.831

Table 10. The results for reviewer features of the electronic product domain using SVM

Reviewer Features	Accuracy	Precision	Recall	F-Measure	AUC
F19, 20	0.734	0.783	0.769	0.776	0.725
F19, 20, 21, 22	0.762	0.787	0.827	0.807	0.746
F19, 20, 21, 22, 18	0.778	0.793	0.852	0.822	0.760
F19, 20, 21, 22, 18, 23	0.781	0.795	0.856	0.824	0.763
F19, 20, 21, 22, 18, 23, 24	0.782	0.797	0.854	0.824	0.764
F19, 20, 21, 22, 18, 23, 24, 25	0.790	0.809	0.850	0.829	0.775

Table 11. The results for reviewer features of the book domain using Adaboost

Reviewer Features	Accuracy	Precision	Recall	F-Measure	AUC
F19, 20	0.672	0.805	0.611	0.695	0.732
F19, 20, 21, 22	0.725	0.765	0.793	0.779	0.787
F19, 20, 21, 22, 18	0.727	0.755	0.816	0.785	0.803
F19, 20, 21, 22, 18, 23	0.727	0.755	0.816	0.785	0.803
F19, 20, 21, 22, 18, 23, 24	0.721	0.751	0.814	0.781	0.798
F19, 20, 21, 22, 18, 23, 24, 25	0.720	0.754	0.804	0.778	0.797

Table 12. The results for reviewer features of the book domain using C4.5

Reviewer Features	Accuracy	Precision	Recall	F-Measure	AUC
F19, 20	0.671	0.792	0.625	0.699	0.729
F19, 20, 21, 22	0.725	0.765	0.794	0.779	0.773
F19, 20, 21, 22, 18	0.730	0.758	0.819	0.787	0.787
F19, 20, 21, 22, 18, 23	0.730	0.758	0.819	0.787	0.787
F19, 20, 21, 22, 18, 23, 24	0.733	0.750	0.845	0.794	0.788
F19, 20, 21, 22, 18, 23, 24, 25	0.733	0.751	0.840	0.793	0.787

Table 13. The results for reviewer features of the book domain using SVM

Reviewer Features	Accuracy	Precision	Recall	F-Measure	AUC
F19, 20	0.659	0.741	0.677	0.707	0.654
F19, 20, 21, 22	0.709	0.748	0.789	0.768	0.686
F19, 20, 21, 22, 18	0.701	0.713	0.853	0.777	0.658
F19, 20, 21, 22, 18, 23	0.701	0.713	0.853	0.777	0.658
F19, 20, 21, 22, 18, 23, 24	0.705	0.721	0.844	0.778	0.666
F19, 20, 21, 22, 18, 23, 24, 25	0.698	0.705	0.868	0.778	0.650

Table 8 to Table 13 show that F18, F19, F20, F21, and F22 greatly improve performance, while other reviewer features are less effective. At first, F19 and F20 achieved 78.4% and 73.2%, respectively, on the AUC measure in the electronic product domain and the book domain. When F21 and F22 were added, Accuracy, F-Measure, and AUC improved. F18 also improves performance. Note that when computing feature F19 to F22, the helpful vote and total vote information of the review that is to be classified is not taken into account. Based on the experiment results, the hypothesis that features F19 to F22 reflect the probability that a reviewer writes high-quality reviews is validated; in other words, a reviewer has received many helpful votes, and his reviews are likely in the high-quality classification.

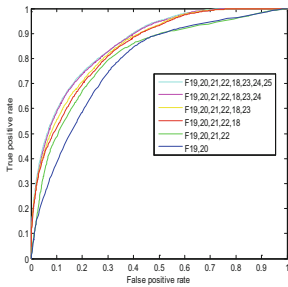


Fig. 11. ROC curves on different reviewer features in the electronic products domain using Adaboost

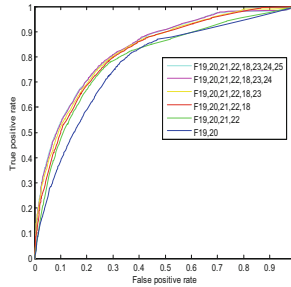


Fig. 12. ROC curves on different reviewer features in the electronic products domain using C4.5

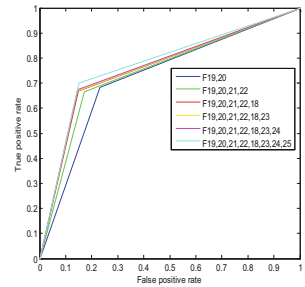


Fig. 13. ROC curves on different reviewer features in the electronic products domain using SVM

Figures 11, 12, 13, 14, 15 and 16 show the ROC curves on different reviewer features using Adaboost, C4.5, and SVM in the electronic product domain and the book domain. It can be seen that features F19 and F20 achieved a relatively high AUC, and the AUC measure increased when features F21 and F22 were added. Reviewer feature F18 also improved the AUC measures, which validates the hypothesis that when a reviewer has published many reviews, his reviews are likely to be in the high-quality classification. Surprisingly, AUC is not improved by feature F23 (reviewer’s domain authority) as much as by F19 to F22.

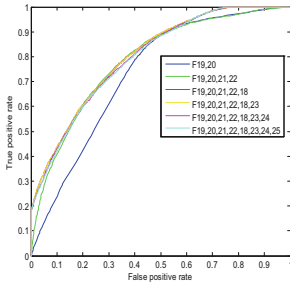


Fig. 14. ROC curves on different reviewer features in the book domain using Adaboost

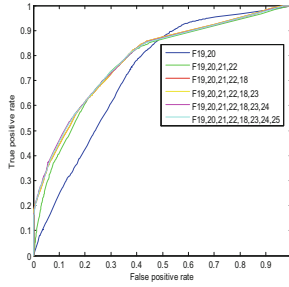


Fig. 15. ROC curves on different reviewer features in the book domain using C4.5

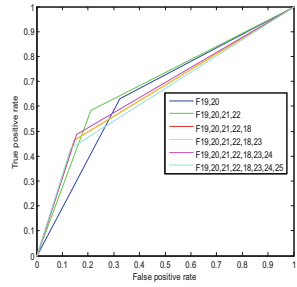


Fig. 16. ROC curves on different reviewer features in the book domain using SVM

5.5 Effectiveness Analysis in Different Domains

From the experiment results in Sect. 5.3, it can be seen that reviewer information indeed matters when predicting review quality in low-quality review detection models. However, reviewer features are more predictive in the electronic product domain than in the book domain. In the electronic product domain, Accuracy, F-Measure, and AUC achieved 83.8%, 86.6%, and 91.3%, respectively, while in the book domain, the best results for Accuracy, F-Measure, and AUC were 74.7%, 80.1%, and 81.0%, respectively. Through analyzing the reviewers' information in the electronic product domain and the book domain, two phenomena were discovered. First, reviewers are likely to write only a few reviews on the electronic product domain, while reviewers in the book domain usually submit many book reviews. The reason for this phenomenon is that consumers usually do not have to purchase many electronic products in the course of a lifetime, while books are usually purchased frequently. Second, reviews in the electronic product domain usually receive more peer-to-peer scoring than reviews on books. The reason for this phenomenon might be that when people are going to purchase electronic products, they often read the reviews for reference and assess the reviews as helpful or not while they are at the review site. However, when people are in the market for books, they seldom read reviews, therefore, book reviews gain less peer-to-peer scoring than electronic product reviews. Based on the conclusion that F21 (the average total votes the reviewer has received) and F22 (the average helpful votes the reviewer has received) are the most effective features; we propose the hypothesis that reviewer information is more effective when the reviews provide a great deal of peer-to-peer information.

6 Conclusion

In this paper, three research questions were considered: 1) Does reviewer information matter for building better models to detect low-quality reviews?, 2) Which reviewer features are most predictive in the detection models?, and 3) How do we acquire and exploit reviewer information to improve low-quality review detection performance? Besides the intrinsic features of the review, some reviewer features are proposed in the low-quality

review detection model. Experiment results on two real world datasets show promising results, from which the following conclusions are derived: First, reviewer information indeed matters when detecting low-quality reviews. Moreover, greater improvement can be achieved when simultaneously using both review features and reviewer features. Secondly, the average helpful votes and the average total votes a reviewer gains from peer-to-peer voting are the most effective features in the reviewer features set. Through effective analysis in different domains, we propose the hypothesis that reviewer information is more effective when the reviewer's reviews provide a great deal of peer-to-peer information. In future work, we plan to apply the proposed model to other user-generated contents.

References

1. Liu, B., Hu, M., Cheng, J.: Opinion observer: analyzing and comparing opinions on the web. In: Proceedings of the 14th International World Wide Web Conference (WWW-2005), 10–14 May 2005, in Chiba, Japan, pp. 342–351 (2005)
2. Popescu, A.-M., Etzioni, O.: Extracting product features and opinions from reviews. In: Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP 2005), pp. 339–346 (2005)
3. Turney, P.D.: Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In: Proceedings of the Meeting of the Association for Computational Linguistics (ACL 2002), pp. 417–424 (2002)
4. Pang, B., Lee, L., Vaithyanathan, S.: . Thumbs up? sentiment classification using machine learning techniques. In: Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP 2002), pp. 79–86 (2002)
5. Kim, S.M., Pantel, P., Chklovski, T., Pennacchiotti, M.: Automatically assessing review helpfulness. In: Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP 2006), pp. 423–430 (2006)
6. Liu, J., Cao, Y., Lin, C.Y., Huang, Y., Zhou, M.: Low-quality product review detection in opinion summarization. In: Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP 2007), pp. 334–342 (2007)
7. Zhang, Z., Balaji, V.: Utility Scoring of Product Reviews. In: Proceedings of the 2006 ACM CIKM International Conference on Information and Knowledge Management (CIKM 2006), pp. 51–57 (2006)
8. Agichtein, E., Castillo, C., Donato, D., Gionis, A., Mishne, G.: Finding high-quality content in Social Media. In: Proceedings of the First ACM International Conference on Web Search and Data Mining (WSDM 2008), pp. 183–194 (2008)
9. Jeon, J., Bruce Croft, W., Ho Lee, J., Park, S.: A framework to predict the quality of answers with nontextual features. In: Proceedings of 29th Annual International ACM SIGIR Conference on Research & Development on Information Retrieval (SIGIR 2006), pp. 228–235 (2006)
10. Zhou, Y., Bruce Croft, W.: Document quality models for Web Ad Hoc retrieval. In: Proceedings of the 2005 ACM CIKM International Conference on Information and Knowledge Management (CIKM 2005), pp. 331–332 (2005)
11. Ghose, A., Ipeirotis, P.G.: Designing novel review ranking systems: predicting the usefulness and impact of reviews. In: Proceedings of the Ninth International Conference on Electronic Commerce (ICEC 2007), pp. 303–310 (2007)

12. Miao, Q., Li, Q., Dai, R.: An integration strategy for mining product features and opinions. In: Proceedings of the 2008 ACM CIKM International Conference on Information and Knowledge Management (CIKM 2008), pp. 1369–1370 (2008)
13. Weimer, M., Gurevych, I.: Predicting the perceived quality of web forum posts. In: Proceedings of the Conference on Recent Advances in Natural Language Processing (RANLP 2007), pp. 643–648 (2007)
14. Chawla, N.V., Bowyer, K.W., Hall, L.O., Philip Kegelmeyer, W.: SMOTE: synthetic minority over-sampling technique. *J. Artif. Intell. Res.* **16**, 321–357 (2002)
15. Yu, P.S., Li, X., Liu, B.: Adding the temporal dimension to search - a case study in publication search. In: Proceedings of the 2005 IEEE/WIC/ACM Conferences on Web Intelligence (WI 2005), pp. 543–549 (2005)
16. Zhang, Z.: Weighing stars: aggregating online product reviews for intelligent e-commerce applications. *Intell. Syst.* **23**(5), 42–49 (2008)
17. Pang, B., Lee, L.: Opinion mining and sentiment analysis. *Found. Trends Inf. Retrieval*, **2**(1–2), 1–135 (2008). <https://doi.org/10.1561/1500000001>
18. Wilson, T., Wiebe, J., Hwa, R.: Just how mad are you? finding strong and weak opinion clauses. In: Proceedings of the Nineteenth National Conference on Artificial Intelligence (AAAI-2004), pp.761–769 (2004)
19. Kim, S., Hovy, E.: Determining the sentiment of opinions. In Proceedings of the International Conference on Computational Linguistics (COLING 2004), pp. 1367–1373 (2004)
20. Hu, M., Liu, B.: Mining opinion features in customer reviews. In Proceeding of the 19th National Conference on Artificial Intelligence (AAAI 2004), pp. 755–760 (2004)
21. Hu, M., Liu, B.: Mining and summarizing customer reviews. In: Proceeding of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2004), pp. 168–177 (2004)
22. Carenini, G., Ng, R.T., Zwart, E.: Extracting knowledge from evaluative text. In: Proceedings of Third International Conference on Knowledge Capture (K-CAP 2005), pp. 11–18 (2005)
23. Su, Q., Xiang, K., Wang, H., Sun, B., Yu, S.: Using pointwise mutual information to identify implicit features in customer reviews. In: Matsumoto, Y., Sproat, R.W., Wong, K.-F., Zhang, M. (eds.) ICCPOL 2006. LNCS (LNAI), vol. 4285, pp. 22–30. Springer, Heidelberg (2006). https://doi.org/10.1007/11940098_3
24. Shi, B., Chang, K.: Mining Chinese reviews. In: Proceedings of the Sixth IEEE International Conference on Data Mining (ICDM 2006), pp. 585–589 (2006)
25. Feldman, R., Fresko, M., et al.: Extracting product comparisons from discussion boards. In: Proceedings of the Seventh IEEE International Conference on Data Mining (ICDM 2007), pp. 469–474 (2007)
26. Ghani, R., Probst, K., et al.: Text mining for product attribute extraction. *ACM SIGKDD Explor. Newslett.* **8**(1), 41–48 (2006)
27. Wang, B., Wang, H.: Bootstrapping both product properties and opinion words from Chinese reviews with cross-training. In: Processings of 2007 IEEE/WIC/ACM International Conference on Web Intelligence, pp. 259–262 (2007)



Advanced Text Mining Methods for Bilingual Lexicon Extraction from Specilized Comparable Corpora

Sourour Belhaj Rhouma^{1(✉)}, Chiraz Latiri^{1(✉)}, and Catherine Berrut^{2(✉)}

¹ University of Tunis El Manar, Faculty of Sciences of Tunis, LIPAH-LR11ES14,
2092 Tunis, Tunisia

sourour.bhr@gmail.com, chiraz.latiri@gnet.tn

² University of Grenoble Alpes, LIG laboratory, MRIM group, Grenoble, France
catherine.berrut@mrिम.fr

Abstract. Recent works rely on comparable corpora to extract efficient bilingual lexicon. Most of approaches in the litterature for bilingual lexicon extraction are based on context vectors (CV). These approaches suffer from noisy vectors that affect their accuracy. This paper presents new approaches which relies on some advanced text mining methods to extract association rules between terms (AR) and extend them to *contextual meta-rules* (MR). In this respect, we propose to extract bilingual lexicons by deploying standard context vectors, association rules and contextual meta-rules. These proposed approaches utilize correlations between co-occurrence patterns across language. An experimental validation conducted on a specialized comparable corpora, highlights a significant improvement of bilingual lexicon based on MR compared to the standard approach.

Keywords: Bilingual lexicon extraction · Information extraction · Text mining

1 Introduction

Over several decades, a lot of effort has been put into creation of lexicons with high coverage, high translation quality and for specific domain. At a later time, the researches were oriented towards the exploitation of *comparable corpora*. With comparable corpora are sets of text collections which cover roughly the same subject area in different languages, but which are not translations of each other. Most of works in BLE task from comparable corpora represent the *standard approach* [1–5] which is based on the context vectors (CV). These vectors store a set of words which are for the neighbourhood of the base word and share the same lexical context. The relation between a base word and its context is called a co-occurrence relation.

Moreover, in text mining (TM) field, one of the main techniques generating knowledge based on co-occurrence relation is association rule (AR) extraction introduced in [6]. These rules provide information on the inter-terms correlations.

For each domain, common language pairs and commercially important subject areas such as medicine, specific dictionaries would be developed. Thus, we encode our intuition into new approaches for extracting bilingual lexicons from specialized comparable corpora. Our proposed approaches relies on enriched representations of the word,

especially those derived through Formal Concept Analysis (FCA) paradigm [7] and some advanced TM methods. These methods are deployed to extract AR and extend them to *contextual meta-rule* (MR). These later capture all the words related to AR associated with the base word. This leads to a less sparse representation. Therefore, we focus on how to compute similarities between AR and between MR using their specific metrics, namely support and confidence. Finally, we compare extracted lexicons using these two patterns with regard to that of the standard approach.

The article is organized as follows: in the Sect. 2, we present the standard approach, their improvements and extended approaches to the task of BLE. Then, we describe, in Sect. 3, our different models for the BLE based on CV, AR and MR. In Sect. 4, we describe our combination strategies applied for extracted lexicons. Section 5 will be dedicated to the linguistic resources and evaluation results of extracted lexicons. The Conclusion section wraps up the article and outlines future works.

2 Related Works to the BLE from Comparable Corpora

Most of the works of the state-of-the-art dealing with BLE task from comparable corpora are based on the standard approach [1–5]. The approaches can be classified in three families, as follows:

1. Standard Approach (SA): The main assumption of the SA states that words with a similar meaning are likely to appear in similar context across languages. Therefore, a word can be represented as a context vector (CV) in source or target language. The dimensions of the source and target vectors are totally different, because each of them is represented by words in a source language and words in a target language. In order to enable the comparison of source and target CV, words in the source CV are translated into the target language using an initial bilingual dictionary. The most popular measure for comparison is the cosine measure, but other authors studied other measures. Thus, we have a list of candidate translations for the base word ranked according to their similarity scores.
2. Improvements of SA: Several contributions have been proposed to improve each step of the SA. [8] combine the information provided by translations context with transliterations¹ and scientific compound words in target language. [9] suggest that CV should be based on the most important contextually relevant words (in-domain terms), and thus propose a method for filtering the noise of the CV. Some researchers looked into adding additional linguistics resources by combining a general dictionary with a specialized dictionary [10] or a multilingual thesaurus [11]. [12] present two techniques for filtering the entries of the initial dictionary (POS-tagging criteria and relative frequency ratio criteria). [4] introduce a word sense disambiguation process that identifies the translations of polysemous words that are more likely to give the best representation of CV in the target language.
3. Extensions of SA: Other approaches have been proposed for BLE that diverge from the SA. In [11], the interlanguage similarity approach avoids the direct translation

¹ Transliterations are words adapted from a source language to a target language, based on their pronunciation.

of the elements of the CV. The principle is to associate to each base word the closest CV to terms in the initial dictionary by using a similarity measure. Other works have used geometric approaches based on two spaces of vectors. The translation step consists of transferring the vectors from the space of the source terms to space of the target terms [13]. We also distinguish the syntactic approach, which is based on the observation that a word and its translation tend to share the same syntactic dependency relations [14]. [15] combine the contextual representation within a thematic one. The assumption is that a term and its translation share thematic similarities. Other work combine two representation of the context, namely a contextual representation (by bag of words) and a syntactic representation (by syntactic dependency relations) [16, 17]. [5] introduce an intermediary step that consists of re-estimating the observed word co-occurrence counts either by smoothing or by prediction techniques. Many single terms are compositional (composed of roots and affixes) and this information can be very useful to match translational pairs, especially for infrequent terms where distributional methods often fail. [18] present the compositional approach based on a proposed bilingual morpheme extraction methods.

Our contribution introduced in this paper, can be seen as an extension of the SA aims at using two new patterns except of context vectors. Our approaches differ from other works in two ways: (1) The new patterns are used, according to our knowledge, for the first time for the task of BLE from comparable corpora. These patterns are based on FCA and some advanced TM methods known as association rules extraction. Association rules are also extended to a mined new pattern named contextual meta-rules which capture all the words related to AR associated with the base word. Comparing to CV, MR leads to a less sparse representation. (2) As only one word representing a rule is usually present in a given context, CV fail to integrate all the words related to a given set of association rules. Therefore, it is easier to adapt our approaches to other language pairs and without any condition about the size of the corpora. In addition, we compare the SA using CV to BLE based on AR, BLE based on MR and their combinations.

3 Proposed Approaches

Our goal is to extract bilingual lexicons using two patterns AR and MR, providing additional and implicit knowledge. The Fig. 1 depicts our proposed approaches: (1) BLE based on AR, and (2) BLE based on MR, it consists in three major steps.

3.1 Formalisation and Definitions

After introducing some notations, we state the definitions of the concepts used in the reminder of the paper. In this respect, Table 1 provides an overview of the notations used in this and later sections.

In this paper, we use in TM field, the theoretical framework of FCA presented in [7]. First, we formalize an extraction context made up of documents and index terms, called *textual context*.

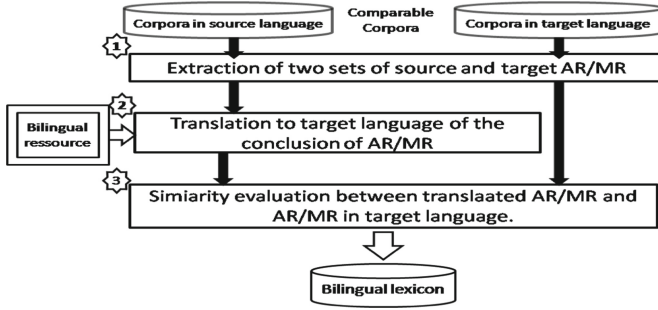


Fig. 1. Overview of our two approaches

Table 1. Summary of notations

Notation	Description	Notation	Description
\mathcal{R}_S	Set of AR in source language	\mathcal{R}_T	Set of AR in target language
\mathcal{MR}_S	Set of MR in source language	\mathcal{MR}_T	Set of MR in target language
AR_S	AR in source language ($AR_S \in \mathcal{R}_S$)	AR_T	AR in target language ($AR_T \in \mathcal{R}_T$)
MR_S	MR in source language ($MR_S \in \mathcal{MR}_S$)	MR_T	MR in target language ($MR_T \in \mathcal{MR}_T$)
P_S	The premise part of AR_S or MR_S	P_T	The premise part of AR_T or MR_T
C_S	The conclusion part of AR_S or MR_S	C_T	The conclusion part of AR_T or MR_T
w_S	A word in source language ($w_S \in \mathcal{C}_S$)	w_S^j	A candidate translation of w_S

Definition 1 (Textual Context). A textual context is a triplet $\mathcal{TC} = (\mathcal{D}, \mathcal{V}, \mathcal{I})$ such as:

- $\mathcal{D} = \{d_1, d_2, \dots, d_p\}$ is a finite set of documents of the corpus.
- $\mathcal{V} = \{w_1, w_2, \dots, w_q\}$ is a finite set of q distinct words of the corpus (i.e., vocabulary of the corpus).
- \mathcal{I} is a binary relation, i.e., $\mathcal{I} \subset \mathcal{D} \times \mathcal{V}$, which connects every document with the words of the corpus which are associated with it.

Definition 2 (Association Rule Between Terms). An association rule (AR) binds two termsets², which respectively constitute its premise (T_1) and conclusion (T_2) parts [6]. Thus, an AR estimates the probability of having the terms of the conclusion (T_2) in a document, given that those of the premise (T_1) are already there.

The advantage of the insight gained through AR is in the contextual nature of the discovered inter-term correlations. Indeed, more than a simple assessment of pair-wise term occurrences, an AR binds two sets of terms, which respectively constitute its premise and conclusion parts.

Given a rule $AR: T_1 \rightarrow T_2$, the support and confidence of AR are computed as follows:

² By analogy to the itemset terminology used in data mining for a set of items.

$$Supp(AR) = Supp(T_1 \cup T_2) \quad (1) \quad Conf(AR) = \frac{Supp(T_1 \cup T_2)}{Supp(T_1)} \quad (2)$$

Definition 3 (Contextual Meta-Rule). A contextual meta-rule, denoted by MR , is an implication of the form: $MR : p \Rightarrow w_1, w_2, \dots, w_k$, such as, $p \in V$ is the premise and $\{w_2, \dots, w_k\} \subset V$ is the conclusion.

The support of a MR , denoted by $Supp(MR)$, is the number of documents $d \in \mathcal{D}$ containing all words $\in \mathcal{V}$ of MR . The support is the minimum of support values of AR selected to construct the MR . The confidence of a MR , denoted by $Conf(MR)$, is the minimum of confidence values of AR constituting the MR . The support and confidence are formally defined as follows:

$$Supp(MR) = \min_{(AR_i \subset MR)} Supp(AR_i) \quad Conf(MR) = \min_{(AR_i \subset MR)} Conf(AR_i) \quad (3) \quad (4)$$

3.2 Extraction of Association Rules and Meta-Rules

In order to extract the most representative terms, a linguistic preprocessing is required on the document collections. The textual context document-terms \mathcal{TC} , is then built by retaining only common nouns, proper nouns and verbs as well the vocabulary \mathcal{V} . The rationale for this focus is that nouns are the most informative grammatical categories and are most likely to represent the content of documents [19]. A stoplist is used to discard functional terms that are very common.

In order to extract AR, we adapted the CHARM-L algorithm [20] to consider any given \mathcal{TC} . The algorithm extracts all the frequent termsets as described in [20], with respect to minimal and maximal support thresholds $minsupp$ and $maxsupp$ ³. These thresholds are experimentally set by considering the *Zipf* distribution of each collection. The construction of a meta-rule, *i.e.*, MR consists of grouping the AR having a common label which is presented by the premise. A meta-rule helps to explain finer relations between terms. Indeed, MR provides a global context for the terms that appear together. New support and confidence values ($Supp(MR)$ and $Conf(MR)$) are defined in Sect. 3.1.

3.3 Translation and Disambiguation of Association and Meta-Rules

In SA, dimensions of the source and target vectors are different from each other and source CV have to be translated in target languages using an initial dictionary in order that the dimensions agree [21]. The core of our approach, as the SA, is the initial dictionary, it allows the translation of AR/MR of a candidate word and compare it to all the target AR/MR to identify the correct translation according to a similarity measure.

For each AR_S or MR_S , for any source word $w_{S_i} \in C_S$, we propose to associate a weight $f(P_S, w_{S_i})$ to assess the relationship of w_{S_i} with the premise P_S . This relation is already given by the confidence value (conditional probability) offered by the AR:

³ *maxsupp* means that the termset must occur at most this user-defined threshold.

$P_S \rightarrow w_{S_i}$. We propose to exploit this relation as a local context of each word $w_{S_i} \in C_S$ with P_S . This generates the construction of the weight vector for each AR or MR in source and target language. A weight vector \vec{V}_R for an AR or MR of the form $P_S \rightarrow w_{S_1}, \dots, w_{S_k}$ is defined as follows:

$$\vec{V}_R = (f(P_S, w_{S_1}), \dots, f(P_S, w_{S_k})).$$

Let us note that each source word $w_{S_i} \in C_S$ is translated to the target language using an initial dictionary. We consider all the translations proposed for a source word by attributing for each the same score f as that of the source word. The word which will have no translation will not be added to the translated AR or MR.

3.4 Similarity Evaluation

Each translated AR or MR is compared to the sets \mathcal{R}_T or \mathcal{MR}_T to filter the most similar ones by using similarity measures. We adopt two methods for computing similarity between AR and between MR respectively:

1. **Context-Based Similarity (CBS):** This method relies on vector representations of word meaning. In SA, the word meanings are represented as vectors into a high dimensional space. In our case, conclusion part of AR or MR are represented as a vector with low dimension. We consider the most commonly used measure which is the cosine [1, 5] of the angle formed by two source and target vector: $Cos(\vec{V}_S, \vec{V}_T) = \frac{\vec{V}_S \cdot \vec{V}_T}{|\vec{V}_S| \cdot |\vec{V}_T|}$.

The similarity between AR or MR is defined as follows:

$$Sim_{context}(AR_S, AR_T) = Cos(\vec{V}_{AR_S}, \vec{V}_{AR_T}) \quad (5)$$

$$Sim_{context}(MR_S, MR_T) = \frac{Cos(\vec{V}_{MR_S}, \vec{V}_{MR_T})}{Conf(MR_T)} \quad (6)$$

$Cos(\vec{V}_{AR_S}, \vec{V}_{AR_T})$ (respectively $Cos(\vec{V}_{MR_S}, \vec{V}_{MR_T})$) are the cosine between the weight vectors of AR_S and AR_T (respectively MR_S and MR_T).

$Conf(MR_T)$ is the confidence value of the target meta-rule MR_T , which leads to show the importance of a MR_T based on its confidence value with a MR_S . We exploit then the *global context* of a MR, namely the support and the confidence, associated for each MR_T .

2. **Semantic-Based Similarity (SBS):** Semantic measures can be reliable because they are based on the judgments of human experts [22]. The semantic similarity between words is assessed based on dictionaries or thesauri. Among the majority of existing semantic similarity measures, WordNet lexical database [23] is used as the underlying resource to calculate semantic similarity. We distinguish six measures using WordNet classified in two categories, namely: (1) Measures based on information content (IC) denoted *RESN* [24], *JCN* [25], and *LIN*. (2) Measures based on path length which simply count the distance between two words in the WordNet taxonomy denoted *PATH* [26], *WUP* [27] and *LCH* [28].

The intuition is to set for each source word w_S , having several translations, the inter-lingual similarity $Sim_{inter}(w_S, C_T)$ of the most similar translation to the words of

the target conclusion part C_T . We compute the semantic similarity Sim_{sem} between two AR or MR by taking the maximum of Sim_{inter} of each source word $w_S \in C_S$. The final formula for computing Sim_{sem} between two AR or MR is defined as follows:

$$Sim_{sem}(AR_S, AR_T) = \max_{w_S \in C_S} Sim_{inter}(w_S, C_T) \quad (7)$$

$$Sim_{sem}(MR_S, MR_T) = \frac{[\max_{w_S \in C_S} Sim_{inter}(w_S, C_T)]}{Conf(MR_T)} \quad (8)$$

The advantage of similarity computing between meta-rules is the exploitation of new values of support and confidence. The semantic similarity is weighted by a score based on the confidence score of the MR_T .

We give an example of AR and MR in source and target language: AR_S : sein \rightarrow traitement (1048; 0.404946), ..., sein \rightarrow cancer (1747; 0.675039).

AR_T : breast \rightarrow cancer (39276; 0.802353), breast \rightarrow study (12922; 0.263978), breast \rightarrow treatment (11339; 0.23164).

MR_S : sein \Rightarrow cas, traitement, risque, cancer, étude, taux, (613 ; 0.236862).

MR_T : breast \Rightarrow cancer, treatment, study, (11339; 0.23164).

The bilingual dictionary is applied on each source term of the conclusion part of MR_S . The translated MR becomes: **sein** \Rightarrow case instance, treatment treating therapy, riskiness risk, cancer, rates rate,. We follow the two similarity calculation strategies between each translated MR and MR_T . For each source premise 'sein', a list of top-ranked translation candidates are selected according to their highest similarities as follows: **sein**: (breast, 2.3245), (aromatase, 2.1607), (chemotherapy, 2.0258), (study, 2.0060), (recurrence, 1.9804), (age, 1.9695), (disease, 1.9591), etc.

To build the lexicon for each approach, we associate for each entry in the source language from AR_S (or from MR_S) a number of top-ranked candidate translations in the target language representing the premises parts of AR_T (or MR_T) and illustrate as follows:

sein breast, aromatase, chemotherapy, study, recurrence, age, disease, etc.

4 Experimental Validation

4.1 Corpora and Linguistic Resources

We evaluate our BLE approaches on a specialized comparable corpora. The Breast Cancer corpus is composed of documents collected from the Elsevier website⁴. The documents were taken from the medical domain within the sub-domain of "breast cancer". The same corpus was used in [5]. The documents have been selected and published between 2001 and 2008 where the title or the keywords contain the term *cancer du sein* in French and *breast cancer* in English. The collected documents counts 130 French documents (about 530,000 words) and 1,640 English documents (about 7.4 million words).

⁴ www.elsevier.com.

The translation is handled using the linguistic resource BabelNet⁵ [29] due to its lexicographic and encyclopedic coverage of multilingual terms resulting from a mapping of the Wikipedia pages⁶ and WordNet, with other lexical semantic resources.

To evaluate the quality of our approaches regards to SA, we built a bilingual reference list. It should be noted that the evaluation of terminology extraction using specialized comparable corpora often relies on lists of a small size: 100 in [3], 125 and 79 in [4]. For Breast Cancer, we selected randomly 170 French words, corresponding to source premises of MR, with their translations in English from the online dictionary *Word Reference*⁷.

4.2 Evaluation

The rank of the correct translation can be considered as an important characteristic when evaluating extracted lexicons. This characteristic is taken into account only by measuring the mean average precision (MAP) defined in [5]. Let n is the number of terms of the reference list, N is the length of candidates translation and r_i is the rank of the correct candidate translation i . If the correct translation does not appear in the top N candidates, $\frac{1}{r_i}$ is set to 0. The MAP is defined as follows:

$$MAP = \frac{1}{n} \sum_{i=1}^N \frac{1}{r_i} \quad (9)$$

Figure 2 shows the different precision values obtained for the standard approach (SA_{CV}), BLE based on Meta-rules (L_{MR}) and BLE based on association rules (L_{AR}). We vary the length of candidate list N from 20 to 500. The low scores obtained for the top 20 candidates are explained that unlike previous work, we do not limit ourselves to very frequent words for evaluation. This is ensured by the step of association rules extraction by setting minimum thresholds of support and confidence. L_{MR} significantly outperforms L_{AR} and SA_{CV} approaches. Indeed, the overall precision of the L_{MR} and L_{AR} approaches increases for large lengths of candidates list as

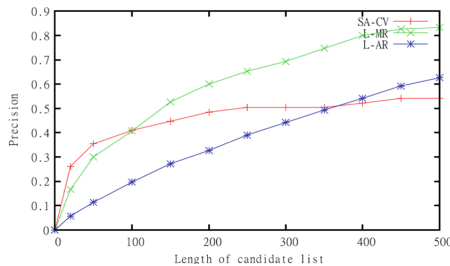


Fig. 2. Precision of SA_{CV} , L_{MR} and L_{AR} for Breast Cancer (FR-EN)

⁵ babelnet.org.

⁶ <http://www.wikipedia.org>.

⁷ <http://wordreference.com>.

$N = 200, 300, 400, \text{ and } 500$. We can see in Fig. 2 that the performance increases by increasing N . This can be explained by the fact that the probability of obtaining correct translations increases with the candidate list growth. While after $N = 150$, performance remains almost constant. In the experiments below, we set $N = 200$. We note that Top_N means that the correct translations of a given word is present in the first N candidates.

4.3 Results and Discussion

Table 2. MAP at Top_{200} with different semantic-based similarity measures

	JCN	LIN	RESN	PATH	WUP	LCH
L_{MR}	0.3503	0.3988	0.3555	0.3895	0.4039	0.3535
L_{AR}	0.2081	0.3836	0.1975	0.2531	0.3885	0.2222

The experients for similarity evaluation adopt two methods for computing similarities as presented in Sect. 3.4. The semantic-based similarity method is performed with respect to the six semantic similarity measures described above. Table 2 displays the obtained results measured in terms of MAP at Top_{200} for scenarios L_{MR} and L_{AR} by varying semantic similarity measures. From these results, we notice that the overall MAP is improved with WUP measure regards to other measures. This increase is important and shows that the similarity between a base word and its candidate translations relies on more information. This information is valuable with semantic measures based on a lexical database.

Table 3. MAP improvement achieved with all scenarios for Breast Cancer. The symbols † indicates statistically significant improvement over the best run in bold, $p - \text{value} < 0.05$

	S_{ACV} (Baseline)	L_{MR}	L_{AR}	L_{CV+MR}	L_{CV+AR}	$L_{CV+MR+AR}$
CBS	0.383	0.4576 †	0.4061	0.4768 †	0.4309	0.5101 †
SBS	-	0.4039	0.3985	0.4527	0.4054	0.4866

Table 3 shows obtained results in terms of MAP. We interpret that our approach based on meta-rules L_{MR} were improve significantly for the three corpora compared to S_{ACV} . This can be explained by the noisy nature of the context vectors with respect to the filtered meta-rules by setting thresholds of support and confidence for their construction. The L_{MR} is better in terms of MAP compared to L_{AR} , and this can be explained by the fact that the rank of the correct translations found for the lexicon based on MR is more important than the correct translations found for the lexicon based on AR. Moreover, the extracted lexicon have a significant improvement with MR deploying Context-based Similarity (CBS) than those deploying Semantic-based Similarity

(SBS) methods. We demonstrate that approaches which combines CV with MR and AR (L_{CV+MR} and L_{CV+AR}) were improved significantly more than L_{MR} and L_{AR} . Besides, approaches which integrates MR, AR and CV ($L_{CV+MR+AR}$) gives the highly significant difference ($p < 0.01$) for Breast Cancer (with a p equal to 0.0059. This can be explained by the fact that the vocabulary used in the breast cancer field is specific and less ambiguous. The obtained results for L_{MR} in term of MAP (45.19%) are better than comparable results reported in [5] (42.3% which is the best MAP assessed for the unbalanced version of the corpus) and those reported in [30] (42.4% for the weighted combinaison).

5 Conclusion and Perspectives

We introduced new approaches for bilingual lexicon extraction considered as an extension of the SA aims at using two new patterns except of context vectors. These patterns are represented by association rules and meta-rules which are extracted based on FCA and some advanced TM methods. We evaluated our approaches on a specialized comparable corpora from the medical domain. More precisely, our different experiments show that using a specialized comparable corpus always improves significantly the quality of extracted lexicons in term of MAP. Moreover, similarity evaluation between meta-rules deploying Context-based Similarity gives the best MAP than those deploying Semantic-based Similarity measures. The results showed that the lexicon based on MR provide a solution for noisy problem, bearing in mind that the contexts are generally limited to few words around the base word, encountered with context vectors. Furthermore, the quality of extracted lexicons combining context vectors, association rules and meta-rules is highly significant. As future work, we first plan to improve the quality of the extracted lexicons. Secondly, we propose a CLIR model based on the extracted lexicons.

References

1. Fung, P.: A statistical view on bilingual lexicon extraction: from parallel corpora to non-parallel corpora. In: Farwell, D., Gerber, L., Hovy, E. (eds.) AMTA 1998. LNCS (LNAI), vol. 1529, pp. 1–17. Springer, Heidelberg (1998). https://doi.org/10.1007/3-540-49478-2_1
2. Rapp, R.: Automatic identification of word translations from unrelated English and German corpora. In: Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics on Computational Linguistics, pp. 519–526 (1999)
3. Morin, E., Daille, B., Takeuchi, K., Kageura, K.: Bilingual terminology mining-using brain, not brawn comparable corpora. In: Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007), Prague, Czech Republic, pp. 664–671 (2007)
4. Bouamor, D., Semmar, N., Zweigenbaum, P.: Towards a generic approach for bilingual lexicon extraction from comparable corpora. In: Proceedings of the 15th Machine Translation Summit, Nice, 2–6 September 2013, pp. 143–150 (2013)
5. Morin, E., Hazem, A.: Exploiting unbalanced specialized comparable corpora for bilingual lexicon extraction. Nat. Lang. Eng. **22**(4), 575–601 (2016)

6. Agrawal, R., Skirant, R.: Fast algorithms for mining association rules. In: Proceedings of the 20th International Conference on Very Large Databases, VLDB 1994, Santiago, Chile, pp. 478–499, September 1994
7. Ganter, B., Wille, R.: *Formal Concept Analysis*. Springer, Heidelberg (1999). <https://doi.org/10.1007/978-3-642-59830-2>
8. Prochasson, E., Morin, E.: Points d’ancrage pour l’extraction lexicale bilingue à partir de petits corpus comparables spécialisés. In: *Traitement Automatique des Langues (TAL)*, vol. 50, pp. 283–304 (2009)
9. Ismail, A., Manandhar, S.: Bilingual lexicon extraction from comparable corpora using in-domain terms. In: Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010), Beijing, China, pp. 481–489 (2010)
10. Morin, E., Prochasson, E.: Bilingual lexicon extraction from comparable corpora enhanced with parallel corpora. In: Proceedings of the 4th Workshop on Building and Using Comparable Corpora, Portland, Oregon, pp. 27–34. Association for Computational Linguistics, June 2011
11. Déjean, H., Gaussier, E.: Une nouvelle approche à l’extraction de lexiques bilingues à partir de corpus comparables. In: Véronis, J. (ed.) *Lexicometrica, Alignement lexical dans les corpus multilingues*, pp. 1–22 (2002)
12. Hazem, A., Morin, E.: Adaptive dictionary for bilingual lexicon extraction from comparable corpora. In: Chair, N.C.C., et al. (eds.) Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC 2012), Istanbul, Turkey. European Language Resources Association (ELRA), May 2012
13. Gaussier, E., Renders, J.M., Matveeva, I., Goutte, C., Déjean, H.: A geometric view on bilingual lexicon extraction from comparable corpora. In: Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL 2004), pp. 526–533 (2004)
14. Yu, K., Tsujii, J.: Bilingual dictionary extraction from Wikipedia. In: Proceedings of NAACL HLT 2009, pp. 121–124 (2009)
15. Rubino, R., Linarès, G.: Une approche multi-vue pour l’extraction terminologique bilingue. In: CORIA 2011, pp. 97–111 (2011)
16. Andrade, D., Matsuzaki, T., Tsujii, J.: Effective use of dependency structure for bilingual lexicon creation. In: Gelbukh, A. (ed.) *CICLing 2011*. LNCS, vol. 6609, pp. 80–92. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19437-5_7
17. Hazem, A., Morin, E.: Improving bilingual lexicon extraction from comparable corpora using window-based and syntax-based models. In: Gelbukh, A. (ed.) *CICLing 2014*. LNCS, vol. 8404, pp. 310–323. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-54903-8_26
18. Hazem, A., Daille, B.: Bilingual lexicon extraction at the morpheme level using distributional analysis. In: Chair, N.C.C., et al. (eds.) Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016), Paris, France. European Language Resources Association (ELRA), May 2016
19. Barker, K., Cornacchia, N.: Using noun phrase heads to extract document keyphrases. In: Hamilton, H.J. (ed.) *AI 2000*. LNCS (LNAI), vol. 1822, pp. 40–52. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-45486-1_4
20. Zaki, M.J., Hsiao, C.: Efficient algorithms for mining closed itemsets and their lattice structure. *IEEE Trans. Knowl. Data Eng.* **17**(4), 462–478 (2005)
21. Kim, J., Kwon, H., Seo, H.: Evaluating a pivot-based approach for bilingual lexicon extraction. *Comput. Intell. Neurosc.* **2015**, 434153:1–434153:13 (2015)
22. Lintean, M.C., Rus, V.: Measuring semantic similarity in short texts through greedy pair and word semantics. In: Proceedings of the Twenty-Fifth International Florida Artificial Intelligence Research Society Conference, May 2012

23. Miller, G.A.: Wordnet: a lexical database for English. *Commun. ACM* **38**(11), 39–41 (1995)
24. Resnik, P.: Using information content to evaluate semantic similarity in a taxonomy. In: *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pp. 448–453 (1995)
25. Jiang, J.J., Conrath, D.W.: Semantic similarity based on corpus statistics and lexical taxonomy. *CoRR* (1997)
26. Rada, R., Mili, H., Bicknell, E., Blettner, M.: Development and application of a metric on semantic nets. *IEEE Trans. Syst. Man Cybern.* **19**, 17–30 (1989)
27. Wu, Z., Palmer, M.: Verbs semantics and lexical selection. In: *Proceedings of the 32nd Annual Meeting on Association for Computational Linguistics, ACL 1994*, pp. 133–138. Association for Computational Linguistics (1994)
28. Leacock, C., Chodorow, M.: Combining local context and WordNet sense similarity for word sense identification. In: *WordNet: An Electronic Lexical Database*. MIT Press (1998)
29. Navigli, R., Ponzetto, S.: Babelnet: building a very large multilingual semantic network. In: Hajic, J., Carberry, S., Clark, S. (eds.) *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL 2010)*, pp. 216–225. The Association for Computer Linguistics (2010)
30. Chebel, M., Latiri, C., Gaussier, E.: Bilingual lexicon extraction from comparable corpora based on closed concepts mining. In: *Advances in Knowledge Discovery and Data Mining - 21st Pacific-Asia Conference (PAKDD 2017)*, 23–26 May 2017, pp. 586–598 (2017)



Classifiers for Yelp-Reviews Based on GMDH-Algorithms

Mikhail Alexandrov^{1,2(✉)}, Gabriella Skitalinskaya³, John Cardiff³, Olexiy Koshulko⁴,
and Elena Shushkevich³

¹ FRUCT Association, Helsinki, Finland
malexandrov.UAB@gmail.com

² Autonomous University of Barcelona, Barcelona, Spain

³ Institute of Technology Tallaght, Dublin, Ireland
gabriellasky@icloud.com, john.cardiff@tudublin.ie,
e.shushkevich@yandex.ru

⁴ Glushkov Institute of Cybernetics, Kyiv, Ukraine
koshulko@gmail.com

Abstract. Yelp is one of the most popular international web resources about products and services that provide users with useful information on local businesses and helps the business owners to make their business more attractive for the users. The Yelp dataset consists of attributes for describing the business, reviews in free text form and numeric star ratings out of 5. The utility of such a dataset has provoked dozens of publications related to classifiers of ratings, which used various smart tools of opinion mining. Unlike them, in this paper we propose to use simpler approaches, namely: (a) selection of descriptors based on term specificity, and (b) formation of classifiers with these descriptors based on inductive modeling. The latter is implemented by the well-known tool GMDH Shell, where GMDH stands for Group Method of Data Handling. This method allows us to build models with high noise immunity. We compare 96 prediction models with identified descriptors by combining various variants: (i) preprocessing with data transformation and balancing classes, (ii) algorithms of classification; and (iii) post processing with ensembling. Instead of the typical 5- star classification we consider combined classes reflecting a more practical view on purchase of goods or development of business. The experiments refer to the most popular categories of business: restaurants and shopping. To evaluate the quality of classifiers we consider the results of predecessors, and we also introduce the so-called defensible accuracy. With this comparison the results presented in the paper prove to be promising.

Keywords: Yelp · GMDH · GMDH shell · Text mining · Opinion mining

1 Introduction

1.1 Motivation

Our motivation for undertaking this research is to answer several questions, namely: why does the study of forums like Amazon, Epinions and Yelp prove to be important

for business development? Why do owners of business take into account these forums? Why do users prefer to post their notes about products and services?

Numerous studies (see, e.g. [6]) clearly show that forums have a significant impact on consumer purchase decisions as well as on business revenues. The same studies also show that there are no direct relations between a given review and a given star rating. So, one needs to generalize the existing information using dozens or even hundreds of reviews concerning a given business or similar businesses. Obviously, this procedure is time consuming and it needs application of effective computer tools.

Many small business owners are motivated to start their own business not only because they want to have any material benefits, but also because they want to change their way of life, as well as other personal factors. The study [40] shows that non-financial objectives can lead to alternative measures of success, especially in the small business sector. This trend is based on the fact that all financial characteristics indirectly imply that a company would like to grow and to increase their capital. However, some companies are not interested in growth and it means that financial indicators such as profit are not their main and the only motivation. Therefore here business owners have other non-financial criteria to measure their success. One of such factors is the personal satisfaction of these business owners. The personal satisfaction and achievements along with the pride for work and a free way of life are often valued higher than material goods.

Here we can ask a question about the other opportunities to know client opinions. For example, these companies could use their own sites. This situation was studied in [14] with the following conclusions: socially-oriented sites are considered trustworthy, and the opinions presented in these sites are considered impartial. The effect of trust in socially-oriented sites has a strong influence on the involvement of users to on-line activities, while non-social sites do not have a significant impact.

Users also have strong motivation to communicate their opinions on the sites related to products and services. We could mention here the following motives [10]: a) a consumer wants to contribute to a community by posting his/her own review and comments on products and services being interesting to other members of the community; b) a consumer feels satisfaction when other participants of a given Internet platform approve his/her contribution (such a feedback can be formal from platform operators or informal from other users); c) a consumer has a complaint against a certain company, and the Internet platform facilitates the presentation of complaints (it is possible if there is a third party - the moderator of the system, which communicates with the company on behalf of the client).

1.2 Related Work

Opinion mining Yelp dataset

The Yelp dataset [40] contains information about approximately 1 million businesses. Each business is described: a) formally by means of its attributes reflecting its location and functionality; and b) informally on the basis of reviews and star assessments. This dataset is used in numerous applications concerning business of products and services. We consider only those related to text processing.

The Yelp dataset contains many omitted and noisy data. For these cases, the authors of [9] propose a generic approach to factorization of data that jointly models relations in the Yelp database. Here ‘data’ and ‘relations’ mean reviews, attributes, and categories of business. Having revealed a set of factors that are shared across existing data the model is able to reflect the information about other relations. The paper also presents joint visualizations of factors being built on terms, attributes and categories and shows some dependencies between them that are not directly observed in the data.

The authors of [26] also deal with hidden factors but in this case, they reveal them in the framework of the problem of personalization. The authors build hidden factors and join them to hidden topics, which are revealed using LDA (Latent Dirichlet Analysis). This approach allows yields: a) easy interpretable textual labels for latent rating dimensions, which helps to ‘justify’ ratings with text; and b) discovered topics that can be used to facilitate solutions to other problems related to automated genre determination, and to identify useful and representative reviews. The examples refer to Amazon and Yelp collections.

Traditional topic modeling lacks methods of incorporating star ratings or semantic analysis in the generative process. The author of [21] proposes a modified LDA, in which term distributions of topics are conditional on star ratings. It is easy to see that such an approach reflects personalization of solutions. The author shows that where one examines topic mixture in documents then this approach produces clearer and more semantically-oriented topics than those of traditional LDA. The experiments use the Yelp dataset [41].

The paper [22] refers to the problem of text annotation. The authors propose a new method for extracting quality phrases from text corpora integrated with phrasal segmentation. This method requires only limited training but the quality of generated phrases proves to be close to human judgment. The method is scalable: both computation time and required space grow linearly as the corpus size increases. The experiments are performed on Academia and Yelp collections.

Yelp restaurant reviews is the subject of consideration in [11]. The authors use on-line LDA to discover latent subtopics on the basis of a large amount of reviews with high dimensionality. These subtopics can provide meaningful insights to restaurants about the needs of customers in order to increase their Yelp ratings, which directly affects the revenue of restaurant owners. The paper shows the breakdown of hidden topics over all reviews, predicts stars per hidden topics discovered, and extends our findings to that of temporal information regarding restaurants peak hours.

The authors of the paper [4] attack the problem of predicting a user’s star rating from two parts: a) extraction of the best set of features from given texts, and b) choice of the best machine learning algorithm. The former uses 4 feature extraction methods: (i) unigrams, (ii) bigrams, (iii) trigrams, and (iv) latent semantic indexing. The latter uses 4 machine learning algorithms: (i) logistic regression, (ii) Naive Bayes classification, (iii) perceptrons, and (iv) linear Support Vector Classification. Taken together these variants form 16 models for predicting the ratings from reviews. The authors analyze the performance of each of these models on Yelp dataset and propose the best one.

This publication gives a good baseline for those who deal with various problems of classifications on Yelp dataset.

Classification of texts with GMDH

GMDH (Group Method of Data Handling) is a method of machine learning, which allows us to build models of optimal complexity from a given class of models to describe experimental data [8, 27, 34]. Due to its universality the polynomials of many variables prove to be the most popular class in many applications. In particular, this class is used in the well-known tool GMDH Shell [31] and also in the majority of tools presented in [32]. Hereinafter we consider only GMDH applications related to classification of textual data.

The paper [2] demonstrates the application of the GMDH based technique for building empirical formula to evaluate politeness, satisfaction and competence reflecting in dialogs between passengers and Directory Inquiries of a railway station in Barcelona. The formulae contain sets of linguistic indicators preliminary assigned by experts separately for each mentioned problems (politeness, satisfaction and competence).

In [3], the authors present the results of building opinion classifiers for Peruvian Facebook, where users discuss the quality of various products and services. The authors use a) linguistic indicators prepared by experts, which automatically form two variables that determine the contribution of positive and negative units, and b) GMDH Shell tool [31] for the selection of an optimal model in the class of polynomial models including the mentioned variables. Each formula reflects the contribution of positive/negative units in a text. The total accuracy reached in the experiments significantly improved the results obtained by other researchers.

The paper [16] demonstrates the possibility of building a classifier of primary medical records using GMDH Shell. The linguistic indicators are extracted from training data sets related to six stomach diseases. The accuracy of results on a real corpus of medical documents proved to be close to 100%. Such a result essentially exceeded the results of other methods, which had been used on the same data set.

In the paper [18], one builds classifiers of texts reflecting opinions of currency market analysts about euro/dollar rate. The classifiers use various combinations of classes: growth, fall, constancy, not-growth, not-fall. The process includes term selection based on criterion of term specificity and model selection using the technique of inductive modeling. The latter is implemented with the GMDH Shell tool mentioned above. The experiments evaluate quality of classifiers and their sensibility to term list. This work has an essential practical orientation.

Authors of the paper [1] test different classifiers from GMDH Shell on small and noisy samples of texts from the popular health-related forum Askpatient. The results show that in dealing with such datasets, GMDH Shell is superior to other well-known methods of machine learning from scikit learn library including neural network LSTM.

1.3 Problem Setting

The review presented above defines the contents of the paper: we study possibilities to predict star rating using less smart and more interpretable methods. To select terms we use the procedure based on criteria of term specificity. Such a criterion compares the

relative frequency of term occurrence in a given corpus and in some standard corpus. To build the model itself we use the Group Method of Data Handling (GMDH). This method (it is better to say ‘technology’) evaluates step by step models of a given class from the simplest ones to the more complex ones to provide the best noise immunity. Neither criterion of term specificity nor GMDH have been used before in the problem under consideration.

Unlike the typical 5 star rating for 5 categories of success we consider classification on 2 classes and 3 classes; each of them is a certain combination of several star categories. These classifications are easily interpretable and allow to take into account extreme classes and relatively successful classes.

We test the values of the mentioned criterion of specificity and select a compromise between the limited and redundant term lists. Here we take into account that the less informative terms will be then automatically filtered by GMDH. To build the best model we combine different options of preprocessing and post processing, and also test several methods of classification. The best model is selected separately for grouping on 2 classes and 3 classes. Totally we deal with 120 variants. Term selection and all procedures of classification are completed by the program LexisTerm [23] and the package GMDH Shell [31].

To evaluate the results of experiments we take into account not only the results of previous research, but also the so-called defensible accuracy. We introduce this notion to take into account the coincidence of expert opinions concerning star rating. Naturally, if experts have different opinions then we should not try to reach the accuracy of 100%. Unfortunately, this circumstance is not taken into account in applied research with subjective human opinions.

In the experiments we use two datasets of 1000 objects each taken from the Yelp resource [41]. The first one is related to restaurants and the second one is related to shopping. Restaurants and shopping are the most popular topics on Yelp. These two samples are the representative ones from the point of view of object distribution between different star categories.

The paper is organized as follows. Section 2 describes linguistic resources. Section 3 introduces classifications used in the research. Section 4 presents the technique of modeling. Section 5 contains the results of experiments. Section 6 concludes the paper.

2 Data Description

2.1 Representativity of Samples

The Yelp dataset contains approximately 4 million reviews about activity of 0.1 million companies. These companies reflect about 1,000 types of business. The general characteristics of Yelp dataset presented in Table 1 relate to the 5 most numerically popular business categories [4]. It should be noted that some reviews reflect opinion about several aspects of the same business simultaneously (e.g. shopping and food). For this reason the total % of reviews exceeds 100% of these 5 business categories.

Table 1. Distributions of companies and reviews [%].

No	Categories	Companies	Reviews
1	Restaurants	34	68
2	Shopping	15	6
3	Food	12	13
4	Home services	12	10
5	Beauty	8	4

In the experiments we consider reviews related to restaurants and shopping as they are the most popular. The selection of 1,000 documents was completed by a random way. To test the representativity of the selected data we compared the star distributions for the category Restaurants based on Yelp [4] with our sample. The results are presented in Table 2. One can see the closeness of both distributions, so our sample can be assumed to be representative.

Table 2. Star distributions [%].

Dataset	5*	4*	3*	2*	1*
Yelp (restaurants)	33	33	16	10	8
Sample (restaurants)	27	34	19	10	10
Sample (shopping)	34	28	13	9	16

We have no data concerning the star distribution for the category Shopping based on Yelp. To avoid this difficulty we compare the star distributions for restaurants and shopping on our samples. These distributions are expected to be similar due to the similar needs of users with respect to these services. Table 2 shows the closeness of these distributions and this circumstance can be an indirect proof that the sample for the category Shopping can be also assumed to be a representative sample.

2.2 Term Selection

Two approaches to term selection

The review-based rating prediction is a problem of sentiment analysis, which relies on different descriptors extracted from a given text/corpus. By ‘descriptor’, we mean a term or term combination, whose frequencies are used in predicting a model. There are two different approaches used for descriptor selection: lexical (lexicon-based) approach [39] and machine learning approach [29].

The lexical approach is based on semantic orientation (SO) lexicons (words with their semantic orientation) and calculates an overall sentiment by aggregating the values of those words presented in a text or a sentence. The classical example of lexical

approach is the tool SO-Calc widely used in many applications. Its vocabularies contain approximately 5,500 terms distributed between 4 vocabularies (nouns, verbs, adverbs, and adjectives together with intensifiers) [38]. The integer SO value assigned to each term varies between -5 and 5 and the sum of these values determines the polarity of opinion of a document under consideration. The SO-Calc vocabulary was tested in [15] using various scales of SO: between -3 and 3, between -2 and 2, and the binary one $\{-1, 1\}$. With the binary scale the authors reached accuracy 0.78 vs 0.83 with the SO-Calc vocabulary. It means that it is not necessary to use too fine-grained scales.

The machine learning approach uses collections of labeled texts as training data in order to build automated classifiers. The machine learning approach is presented in the well-known comprehensive survey [30]. The majority of cases presented in this survey concern binary classification. The authors working with rating prediction use more smart ways. For example, in [20] one builds a vocabulary of sentiments using term strength with respect to each of 5 classes. The term strength is determined by the relative frequencies of term occurrence in these classes. Then this vocabulary is used in collaborative filtering algorithms.

Criterion of term specificity

In our work we use a combined approach consisting of two phases. Initially we build a domain-oriented lexicon using criterion of term specificity. Obviously this phase refers to a lexical approach. Then we select the most informative terms in the process of inductive modeling. The latter is one of the technologies of machine learning.

By ‘Term Specificity’ with respect to a given corpus we mean a factor $K \geq 1$, which shows how much term frequency in the corpus $fC(w)$ exceeds its frequency in any standard corpus $fL(w)$: $K = fC(w) / fL(w)$. In our work we use the General Lexis of English that reflects term frequencies in the British National Corpus. This lexis is available online [17]. It is easy to see that the higher K is, the less number of terms is selected. The experience shows that in all cases when $K \geq 3$ then both stop words and almost all general lexis are eliminated. So, users may not think about this kind of words.

We built lists of terms for $K = 2, 5, 10, 20, 50$ separately for the categories Restaurants and Shopping, in total 8 lists. Then the clearly unuseful words were removed from each list. We used here the logarithmic step for variation of K to obtain essential changes in these lists. It was found that: when $K = 2$ the lists included many insignificant terms, when $K > 5$ we lost many useful terms. So, we took the threshold $K = 5$ as the compromise. Table 3 shows the sizes of some lists after and before correction (in parenthesis). All examples are presented as stems. One can see that when the factor K changes on logarithmic scale the size of lists changes in lineal (almost lineal) scale. It is typical for different domain-oriented corpus of documents.

Table 3. Sizes of lexicons.

Dataset	$K = 5$	$K = 10$	$K = 20$	Examples (stems)
Restaurants	120 (142)	68 (70)	33 (33)	Amaz beef bowl cool ...
Shopping	127 (160)	41 (46)	17 (18)	Amaz bike brand dress ...

The criterion of term specificity is realized in the program LexisTerm, which is described in detail in [23]. This program was used in our projects in Peru, Spain, and Russia [3, 16, 18]. We also used it in this research.

It should be noted that the criterion of term specificity proves to be very useful not only for analysis of Yelp reviews. It can be also useful for many other applications, where topic-oriented vocabularies are necessary for numerical presentation of documents. The typical way for building such vocabularies with the mentioned criterion consists of standard steps:

1. Experts in a given area present a representative set of documents related to this area. Speaking of the total volume of documents (number of their words) one should take into account one remarkable regularity, namely: the number of new terms increases in arithmetic progression, when the total number of terms increases in geometric progression. This lexical regularity is firstly described in [37].
2. A computer linguist builds a topic oriented vocabulary using the criterion of term specificity. He/she has the possibility to manage the depth of topic presentation by changing the factor K . Therefore we have here the scalable method of topic presentation.

By the way if one knows in advance the approximate number of subtopics in the framework of a given topic (from... to ...) then it is possible to build vocabularies for these subtopics simultaneously with the vocabulary for the whole topic. For this it is necessary to use the criterion of term specificity and technique of clustering. This method is described in [25].

2.3 Parameterized Documents

1,000 documents of the category Restaurants and 1,000 documents of the category Shopping mentioned above are presented in vectorial form in the space of selected terms. Therefore we have two matrices document-term containing term frequencies. The characteristics of matrices are shown in Table 4. Here: 'Dimension' reflects the number of documents and terms, 'Max freq.' is equal to the maximum number of term occurrences, 'Zero %%' shows the percent of zero values in matrices. The other data is the number of documents having a given rank. Obviously, the values 1* and 5* mean the worst and the best points.

Table 4. Characteristics of matrices.

Dataset	Dimension	Max freq	Zero %%	No. 5*	No. 4*	No. 3*	No. 2*	No. 1*
Restaurants	1000 x 120	18	90	267	344	193	97	99
Shopping	1000 x 127	14	97	346	276	133	88	157

We also measured the completeness of document images, that is the number of key-terms the images include. The results are presented in Table 5. Here: ‘Aver. Number’ is the average number of terms in documents of a corpus under consideration.

Table 5. Number of key-terms in documents.

Dataset	Min number	Max number	Aver. Number
Restaurants	1	91	15.6
Shopping	1	69	9.9

These tables show that the matrices are very sparse that worsens the results of classification. For this reason in our next research we plan to enrich the lists of terms with the most significant 2-g and 3-g of letters. For selection of these k-grams the same criterion of term specificity is supposed to be used.

3 Classifications and Its Quality

3.1 Combined Classes

The assessment of a business in Yelp-reviews is evaluated by means of ‘stars’. They reflect rank classification from 1* to 5*. For this reason the quality of automatic classifiers in publications is also evaluated on this scale (see, for example [3, 8]). On the other hand, such a 5-class scale seems to be too detailed for real applications, when it is necessary only to say whether a given business is successful/unsuccessful or whether this business is extreme/satisfactory. For this reason, in this paper we use combined classes that are better oriented on practical situations. The corresponding classifications are presented in Table 6 using the 5-class scale.

Table 6. Combined classes.

Contents	Success of companies	Class 1	Class 2	Class 3
2 classes	Unsuccessful, others	1*, 2*	3*, 4*, 5*	
3 classes	Failed, satisfactory, excellent	1*	2*, 3*, 4*	5*

With the data from Tables 4 and 6, we can calculate new distributions of documents between combined classes. They are presented in Tables 7 and 8. It is easy to see that the distribution between combined classes proves to be essentially more unbalanced in comparison with the distribution between initial star-classes. So, we carefully tested the option of balancing classes in the process of modeling (see Sect. 5).

The proposed classification on 2 combined classes may be useful when one needs to avoid unsuccessful purchase or unsuccessful development of his/her business. The

Table 7. Distribution of documents on 2 combined classes (unsuccessful-other).

Dataset	Class 1 (1*,2*)	Class 2 (3*,4*,5*)
Restaurants	196 (20%)	804 (80%)
Shopping	245 (25%)	755 (75%)

Table 8. Distribution of documents on 3 combined classes (failed-satisfactory-excellent).

Dataset	Class 1 (1*)	Class 2 (2*,3*,4*)	Class 3 (5*)
Restaurants	99 (10%)	634 (63%)	267 (27%)
Shopping	157 (16%)	497 (50%)	346 (34%)

proposed classification on 3 combined classes may be useful when we are ready to purchase something with satisfactory quality or to remain with a satisfactory level of business having avoided the extreme situations.

3.2 Defensible Accuracy

In the paper we propose an approach for building classifiers related to Yelp-reviews. These classifiers are tuned to the new classes more oriented on practical needs of customers and businessmen. The question is how to evaluate the quality of classifiers and how to evaluate their advantage?

The quality of classifiers can be assessed with the different measures described in the well-known books on Information Retrieval [5, 24]. The typical approach is a comparison of results of classification with a given Gold Standard (GS). The most popular measure here is so-called group F-measure introduced in the paper [32]. This measure takes into account the values of recall and precision for each class and the sizes of these classes. In case of classification (unlike clustering) it can be written as follows:

$$F = \sum_i (n_i/N) \max_j (F_{ij})$$

Here: $i = \{1,2,\dots,m\}$ is the counter for classes in GS; $j = \{1,2,\dots,k\}$ is the counter for classified groups (clusters); m is the number of classes; k is the number of groups (clusters); n_i is the number of documents in class i ; N is the total number of documents; F_{ij} is a partial F-measure for group j with respect to class i .

The other well-known measure is so-called A-measure, that is accuracy. It is a very simple and essentially less smart measure than F-measure. Usually accuracy takes into account the relative number of successful cases of classification independently of classes:

$$A = n/N$$

Our past experience shows that both measures are approximately equal when we deal with hundreds or even dozens of objects and a small number of classes. So, we use A-measure having in view this circumstance.

To evaluate the advantage of the proposed classifiers we should compare our results with the others obtained before us. But here we meet the principal difficulty: our classifiers use combined classes described above but all our predecessors dealt with the 5-star classification. In order to cover such a problem we introduce so-called *defensible accuracy*, which shows the upper level of accuracy to be reached by the method under consideration. The fact is we consider opinions taken from social networks. The authors of blogs and posts from social networks are only users but not experts in the area. So, their opinions can't be accepted as the final truth. Moreover even experts in the area often have different opinions concerning the same subject or event although the difference between their opinions has less variation. Therefore, when we want to assess any classifiers for social networking then we need to take into account this uncertainty.

To evaluate the defensible accuracy we need:

1. To assess the same set of reviews by several experts
2. To assess the concordance between experts.

The relative number of fully coincident assessments just defines the defensible accuracy. Such a term means that any accuracy higher than the defensible accuracy has no sense because experts have different opinions with respect to a given object or event. In our case these objects or events are given reviews.

The concordance of an expert's opinions means the consistency of a given group of experts. If this group is not concordant then the obtained defensible accuracy is unreliable.

For the experiments we selected 50 reviews of the category Restaurants and 50 reviews of the category Shopping. The star distribution in each mini datasets corresponded to star distribution in the entire dataset of 1,000 reviews reflected in Table 4. These reviews were additionally evaluated by two experts with native English (co-authors of the paper). As an example of data processing we show assessments concerning category Restaurants in Table 9. Here: 'Yelp' is rating according to Yelp dataset, 'Exp1' and 'Exp2' are ratings of two experts mentioned above, 'Class' is combined class according to 2-class grouping, 'Equality: Rating / Class' reflects the coincidence of opinions about rating and class among the experts. Class A is equal {3*,4*,5*}, class B is equal {1*,2*} Coincidence is marked by '1' in case of full coincidence and '0' in other cases.

To calculate the coincidence of expert opinions we used 20 reviews from the mentioned 50 reviews for the work with 2 classes, and the other 30 reviews for the work with 3 classes. The results are presented in Table 10. The data from this Table can be considered as the upper level of the defensible accuracy.

To check the concordance of experts we applied the criterion of Kendall that is the criterion of rank correlation. Table 11 contains the values of this criterion. The 5%-threshold related to Kendall criterion is equal $T = 0.32$ for 20 reviews and $T = 0.25$ for 30 reviews. Because all values exceed these thresholds then Yelp-expert, Expert-1 and Expert-2 belong to the one coordinated group of experts, which we can trust to.

Here: 'Exp' stands for Experts and 'cl' stands for classes.

Table 9. Assessments for documents concerning restaurants (examples).

Examples of documents	Yelp/ Class	Exp1/ Class	Exp2 / Class	Equality: Rating/ Class
Good selection of different poutines but otherwise nothing special. Fries weren't the greatest and couldn't taste the pepper sauce. Plus is that it's open 24h for those times when you got the munchies post bar hopping	2/B	2/B	2/B	1/1
It's an always open creative greasy spoon, popular with after hours crowds who are jonesing for late night eats. There aren't many 24/7 places, and this one is a Montreal staple. Personally, I'd rather eat a bag of Doritos	2/B	1/B	3/A	0/0
Food was by far the worst tasting Mexican 'food' I've ever had. The chips and salsa were flavorless and my burrito had noodles in it! Topped off by the fact that I immediately got sick after eating the vegetarian burrito. I would not recommend this establishment. I can't believe they're still open for business	1/B	1/B	1/B	1/1

Table 10. Coincidence of expert opinions (3 experts).

Category	2 classes	3 classes	5 classes
Restaurants	95%	83%	68%
Shopping	97%	85%	64%

Table 11. Values of Kendall coefficient of concordance.

Category	Exp 1-2 (2 cl.)	Exp 1-3 (2 cl.)	Exp 2-3 (2 cl.)	Exp 1-2 (3 cl.)	Exp 1-3 (3 cl.)	Exp 2-3 (3 cl.)
Restaurants	1.00	0.99	0.99	0.92	0.88	0.96
Shopping	1.00	1.00	1.00	0.95	0.93	0.93

4 Method and Tools

4.1 Group Method of Data Handling (GMDH)

To build classifiers we use the technique of inductive modeling presented by the Group Method of Data Handling (GMDH), developed by the remarkable Ukrainian scientist O. Ivakhnenko. His first International publications related to GMDH appeared in the 1970s [12, 13], but in spite of the long history of GMDH it is still not well-known to researchers. So, we give here a brief description of GMDH:

1. An expert defines a sequence of models, from the simplest to more complex ones.
2. Experimental data are divided into two datasets: training data and control data
3. For a given kind of model, the best parameters are determined with training data using any internal criterion
4. This model is tested on control data using external criteria
5. The external criteria (or the most important one) are checked on having reached a stable optimum. In this case the search is finished. Otherwise, a more complex model is considered and the process is repeated from step 3.

Naturally, this description is a basic one without details about a partition of the dataset on subsets, or a process of search, or criteria. For example, often the dataset is divided on three datasets: training data, control data, and exam data. The latter is used for testing quality of selected models having in view that control data is only the filter for model selection. Besides, sometimes the search is organized in two directions simultaneously: from the simplest model to more complex ones and from the most complex model to the simpler ones, and so on. These details are reflected in [7]. The full survey concerning the history and perspectives of GMDH is published in [34]. The theoretical basis for the GMDH approach is presented in [35].

The typical form of model presentation is the polynomial one:

$$y = a_0 + \sum a_i x_i + \sum b_{ij} x_i x_j + \sum c_{ijk} x_i x_j x_k + \dots$$

Here: y is a dependent variable, x_i are independent variables, a, b, c are coefficients to be determined. We can use both positive and negative power functions x_m , where $m < 0$, or $m > 0$. If we consider the process on half-infinite or infinite intervals then instead of presented polynomials we may use the systems of classical orthogonal polynomials of Laguerre and Hermite respectively.

GMDH selects models of optimal complexity in the framework of a given class of models and it is the principal property of GMDH technique. By ‘complexity’ we mean the number of parameters for model description. Optimal complexity provides the best noise immunity of models: when noise increases the model automatically becomes simpler and vice versa. Such an effect is considered in [35].

One can find the full information about GMDH development and applications in [32].

4.2 GMDH Shell

GMDH Shell (GMDH-S) is a well-known tool for the following applications:

- time series prognosis (extrapolation),
- function presentation (approximation),
- object classification

Including extended possibilities for visualization of results [30]. GMDH-S employs the technique of GMDH. At present GMDH-Shell includes 2 classical algorithms with their modifications:

- Combinatorial GMDH,
- GMDH-type neural networks.

In our research we use the classification option. For this GMDH-Shell uses the One-vs-All method [28, 36], which reduces multiclass classification to binary classification. Each binary classifier is presented here in the form of an equation of dividing surface. Inductive modeling just allows us to find the equation of optimal complexity in n -dimensional space of linguistic variables which we discussed above in Sect. 2.3.

One can download the trial version of GMDH-Shell and test it using his/her own data [31]. Universities have the possibility to purchase this product free of charge for teaching purposes.

5 Experiments

5.1 Preprocessing, Tuning Models and Post-processing

For the experiments we used GMDH-S mentioned above. It includes the following possibilities for preprocessing:

- data normalization to a given interval, e.g. $[-1.0, 1.0]$ or $[0.0, 1.0]$;
- data transformation with various functions such as square root, cubic root or \arctg to suppress or to strengthen small and large values;
- balancing classes using copying for small classes.

In the process of modeling a user can do the following:

- to select one of GMDH-based algorithms,
- to limit the total model complexity,
- to assign the form of elements in polynomials,
- to define the external criterion.

Speaking about post-processing we mean both various forms of visualization for result presentation and the procedure of ensembling. The latter is averaging a set of the best models selected by GMDH-Shell. The number of models to be averaged is assigned by a user.

In the experiments we used normalization on [0.0, 1.0] and tested data transformation, balancing and ensembling. The external criterion in model selection was 2-fold cross validation. For tuning algorithms we used recommendations [19]. Taken together we considered 96 options of modeling. Table 12 presents the number of variants for each option.

Table 12. Number of variants for different options.

Transform	Balancing	Ensembling	Algorithm	Complexity	Form
3	2	2	2	2	2

Here: Transformation = {without transformation, cubic function, arctg function}; Balancing = {without balancing, with balancing}; Ensembling = {without ensembling, with ensembling}; Algorithm = {classical combinatorial, classical polynomial neural network}; the contents of term ‘Complexity’ depends on the algorithms, it can be the number of members in polynomial or the number of neurons in the model; Form = {quasi-linear, quadratic}.

It is necessary to emphasize that we did not test various document presentations in the experiments although such a presentation essentially affects results (sometimes decisively).

5.2 Building Classifiers

The distribution of documents between classes for this category is presented in Tables 7 and 8. Therefore the baselines for A-measure are equal 80% for 2 classes and 63% for 3 classes respectively for the category Restaurants, and 75% for 2 classes and 50% for 3 classes respectively for the category Shopping. The first experiments shows that a) balancing gives worse results than its absence; b) the neural network of GMDH-S works essentially better than the classical combinatorial algorithm. The best options and results are presented in Table 13. Here: NN stands for neural network, yes (Xm) means the option of ensembling with X the best models (X is selected manually), 1000 is the number of neurons.

The accuracies reached in the experiments are close or exceed those presented in other publications related to opinion mining Yelp review. See, for example [4]. However such a comparison is slightly incorrect because we consider classifications on 2 classes and 3 classes instead of 5 classes. So, to evaluate the results we should take into account other indicators such as defensible accuracy and baseline. With this point of view the results can be assumed as the promising ones, but which should be improved using extended lists of key-terms.

Table 13. Results of modeling for the categories Restaurants and Shopping.

Options	2 classes Restaurants	3 classes Restaurants	2 classes Shopping	3 classes Shopping
Transformation	Arctg	Arctg	No	Cubic
Balancing	No	No	No	No
Ensembling	Yes (18m)	Yes (8m)	Yes (6m)	Yes (10m)
Algorithm	NN	NN	NN	NN
Complexity	1000	1000	1000	1000
Form	Quadratic	Quasi-linear	Quadratic	Quadratic
Results	2 classes Restaurants	3 classes Restaurants	2 classes Shopping	3 classes Shopping
Accuracy	0.91	0,73	0.86	0,71
Defensible accuracy	0.95	0.83	0.97	0.85
Baseline	0.80	0,63	0.75	0,50

6 Conclusion

In the paper we propose the simple technique for classification of reviews from the well-known Yelp dataset [41]. This technique is based on a) key-term selection using term specificity, and b) construction of predictive models using inductive modeling with GMDH. We consider classifications on 2 classes and 3 classes instead of 5 classes related to 5-star rating. We suppose that in many cases these classifications, namely {unsuccessful, others} and {worst, satisfactory, best}, will prove to be more useful for practical applications. We introduce so-called defensible accuracy of results that takes into account the variety of opinions of users with respect to the same materials of social networks. We demonstrate a practical example how to calculate this indicator and we postulate its value as a justified accuracy. We hope this approach will be useful for those who deal with processing data from social networks.

For the experiments we use the well-known tool GMDH Shell and we tune its parameters to build the best model. The results prove to be promising having in view both the simplicity of the proposed technique and the mentioned defensible accuracy.

In the paper we did not test various forms of document presentation. We only applied the way we successfully used many times in our projects. We intend to consider this question in future experiments and we will try to maintain the simplicity of the proposed technique.

References

1. Akhtyamova, L., Alexandrov, M., Cardiff, J., Koshulko, O.: Opinion mining on small and noisy samples of health-related texts. In: Shakhovska, N., Medykovskyy, M.O. (eds.) CSIT 2018. AISC, vol. 871, pp. 379–390. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-01069-0_27
2. Alexandrov, M., Blanco, X., Catena, A., Ponomareva, N.: Inductive modeling in subjectivity/sentiment analysis (case study: dialog processing). In: Proceeding of 3rd International Workshop on Inductive Modeling (IWIM-2009), pp. 40–43. Krynica, Poland (2009)
3. Alexandrov, M., Danilova, V., Koshulko, O., Tejada, J.: Models for opinion classification of blogs taken from Peruvian Facebook. In: Proceeding of 4th International Conference on Inductive Modeling (ICIM-2013), pp. 241–246. Publ. House NAS of Ukraine & Czech Tech. Univ, Kyev (2013)
4. Asghar, N.: Yelp dataset challenge: review rating prediction. CS886 Project Report, arXiv: 1605.05362v1 [cs.CL] 17 May 2016 (2016). <https://arxiv.org/pdf/1605.05362.pdf>
5. Baeza-Yates, R., Ribero-Neto, B.: Modern Information Retrieval. Addison Wesley, USA (1999)
6. Chen, P.Y., Wu, S.Y., Yoon, J.: The impact of online recommendations and consumer feedback on sales. In: Proceeding of International Conference on Information Systems, pp. 711–724 (2003)
7. Pavlov, A., Stepashko, V., Kondrashova, N.: Effective Methods of Model Self-Organization. Publ. House ‘Academ Periodica’ [rus], Kyev (2014)
8. Farlow, S.J.: Self-Organizing Methods in Modeling: GMDH-type Algorithms. 1st edn, Statistics: A series of textbooks and monographs, Book 54, CRC Press, USA (1984)
9. Gupta, N., Singh, S.: Collective factorization for relational data: an evaluation on the Yelp datasets. Homepage, https://www.yelp.com/html/pdf/YelpDatasetChallengeWinner_CollectiveFactorization.pdf (2016)
10. Hennig-Thurau, T., et al.: Electronic word-of-mouth via consumer-opinion platforms: what motivates consumers to articulate themselves on the Internet. *J. Interact. Market.* **18**(13), 39–52 (2004)
11. Huang, J., Rogers, S., Joo, E.: Improving restaurants by extracting subtopics from Yelp reviews. In: CS294–1 Spring 2013: final project I. Homepage, https://www.yelp.com/html/pdf/YelpDatasetChallengeWinner_ImprovingRestaurants.pdf, (2013)
12. Ivakhnenko, A.: Heuristic self-organization in problems of automatic control. *J. Automatica (IFAC)* **3**, 207–219 (1970)
13. Ivakhnenko, A.: Polynomial theory of complex systems. *IEEE Trans. System, Man Cybern.* **1**(4), 364–378 (1971)
14. Karakaya, F., Barnes, N.G.: Impact of online reviews of customer care experience on brand or company selection. *J. Consumer Market. USA*, 447–457 (2010). <https://www.researchgate.net/publication/235316561>
15. Kaurova, O., Alexandrov, M., Ponomareva, N.: The study of sentiment word granularity for opinion analysis (a comparison with Maite Taboada works). *Int. J. Social Media MMM: Monit. Measure. Min.* no.1, Publ. House ‘Konvoj’, Brno, 45–57 (2010)
16. Kaurova, O., Alexandrov, M., Koshulko, A.: Constructing classifiers of medical records presented in free text form. In: Proceeding of 4th International Conference on Inductive Modeling (ICIM-2013), pp. 273–278. Publ. House NAS of Ukraine & Czech Tech. Univ., Kyev (2013). <http://mgua.irtc.org.ua/attach/ICIM-IWIM/2013/4.8%20.pdf>,
17. Kilgarriff, A.: BNC database and word frequency lists. <http://www.kilgarriff.co.uk/bnc-rea-dme.html>

18. Koshulko, O., Alexandrov, M., Danilova, V.: Forecasting euro/dollar rate with Forex news. In: Proceeding of 19th International Conference on Application of Natural Language to Information Systems (NLDB-2014), LNCS, pp. 148–153. Springer, Heidelberg (2014)
19. Koshulko, O., Koshulko, G.: Validation strategy selection in combinatorial and multilayered iterative GMDH algorithms. In: Proceeding of 4th International Workshop on Inductive Modeling (IWIM-2011), pp. 51–54. NAS of Ukraine & Prague Tech. University, Kyev (2011). <http://mgua.irtc.org.ua/attach/ICIM-IWIM/2011/7%20.pdf>
20. Leung, C.W.K., Chan, S.C.F.: Sentiment analysis of product reviews. In: Report Hong Kong Univ, Homepage, <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.100.1549&rep=rep1&type=pdf> (2006)
21. Linshi, J.: Personalizing Yelp star ratings: a semantic topic modeling approach. In: Report of Yale University, Homepage, https://www.yelp.com/html/pdf/YelpDatasetChallengeWinner_PersonalizingRatings.pdf (2014)
22. Liu, J., et al.: Mining quality phrases from massive text corpora. In: Proceeding of International Conference of SIGMOD-2015, Homepage, https://www.yelp.com/html/pdf/YelpDatasetChallengeWinner_MiningQualityPhrases.pdf (2015)
23. Lopez, R., Alexandrov, M., Barreda, D., Tejada, J.: Lexistern – the program for term selection by the criterion of specificity. In: Artificial Intelligence Application to Business and Engineering Domain, vol. 24, pp. 8-15. ITHEA Publ., Rzeszov-Sofia (2011)
24. Manning, C., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press, UK (2008)
25. Makagonov, P., Alexandrov, M., Sboychakov, K.: A toolkit for development of the domain-oriented dictionaries for structuring document flows. In: Data Analysis, Classification and Related Methods, Studies in Classification, Data Analysis, and Knowledge Organization, pp. 83–88. Springer, Heidelberg (2000). https://doi.org/10.1007/978-3-642-59789-3_13
26. McAuley, J., Leskovec, J.: Hidden factors and hidden topics: understanding rating dimensions with review text. In: Proceeding of 7th ACM Conference on Recomm. Systems, Homepage, <http://infolab.stanford.edu/~julian/pdfs/recsys13.pdf> (2013)
27. Madala, H., Ivakhnenko, A.: Inductive Learning Algorithms for Complex Systems Modelling. CRC Press, USA (1994)
28. One-vs-All, Wikipedia, Homepage, http://en.wikipedia.org/wiki/Multiclass_classification
29. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up? sentiment classification using machine learning techniques. In: Proceeding of Conference on Empirical Methods in Natural Language Proc. (EMNLP-2002), pp. 79–86. SIGDAT, Philadelphia (2002)
30. Pang, B., Lee, L.: Opinion mining and sentiment analysis. In: Foundations and Trends in Information Retrieval, vol. 2, no. 1–2, pp. 1–135 (2008)
31. GMDH Shell, Homepage, <http://gmdhshell.com>
32. GMDH, Dept of Inform. Tech. in Induct. Model., NAS of Ukraine, Kyev, Homepage, <http://mgua.irtc.org.ua/>
33. Stein, B., Eissen, S.M., Wibbrock, F.: On cluster validity and Information needs of users. In: Proceeding 3rd IASTED Conference on Artificial Intelligence and Application (AIA-2003), pp. 216–221, Acta Press (2003)
34. Stepashko, V.: Developments and prospects of GMDH-based Inductive Modeling. In: Advances in Intelligent Systems and Computing II (Proc. of CSIT-2017), AISC, vol. 689, pp. 474–491. Springer, Heidelberg (2018)
35. Stepashko, V.: Method of critical variances as an analytical tool of the Inductive Modeling theory. J. Inf. Automat. Sci. Begell House Inc., **40**(3), 4–22 (2008)
36. Tax, D.M.J., Duin, R.P.V.: Using two-class classifiers for multiclass classification. In: Proceeding of International Conference on Pattern Recognition, pp. 1051–1054. IEEE, Quebec, Canada (2002). <http://prlab.tudelft.nl/content/using-two-class-classifiers-multiclass-classification>

37. Tweedie, F.J., Baayen, R.H.: How variable may a constant be? measures of lexical richness in perspective. *Comput. Human.* **32**, 323–352. Kluwer Academic Publishers (1998)
38. Taboada, M., Anthony, C., Voll, K.: Creating semantic orientation dictionaries. In: *Proceeding of 5th International Conference on Language Resources and Evaluation (LREC-2006)*, pp. 427–432. Italy (2006)
39. Turney, P.: Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In: *Proceeding of 40th Meeting of the ACL*, pp. 417–424 (2002)
40. Walker, E., Brown, A.: What success factors are important to small business owners. *Intern. Small Bus. J.* **22**(6), 577–594 (2004)
41. Yelp Dataset, http://www.yelp.com/dataset_challenge (2014)



Research Collaboration Analysis Using Text and Graph Features

Drahomira Herrmannova¹(✉), Petr Knoth², Christopher Stahl¹,
Robert Patton¹, and Jack Wells¹

¹ Oak Ridge National Laboratory, Oak Ridge, TN, USA
{herrmannovad, stahlcg, pattonrm, wellsjc}@ornl.gov

² The Open University, Milton Keynes, UK
petr.knoth@open.ac.uk

Abstract. Patterns of scientific collaboration and their effect on scientific production have been the subject of many studies. In this paper we analyze the nature of ties between co-authors and study collaboration patterns in science from the perspective of semantic similarity of authors who wrote a paper together and the strength of ties between these authors (i.e. how much have they previously collaborated together). These two views of scientific collaboration are used to analyze publications in the TrueImpactDataset [11], a new dataset containing two types of publications – publications regarded as seminal and publications regarded as literature reviews by field experts. We show there are distinct differences between seminal publications and literature reviews in terms of author similarity and the strength of ties between their authors. In particular, we find that seminal publications tend to be written by authors who have previously worked on dissimilar problems (i.e. authors from different fields or even disciplines), and by authors who are not frequent collaborators. On the other hand, literature reviews in our dataset tend to be the result of an established collaboration within a discipline. This demonstrates that our method provides meaningful information about potential future impacts of a publication which does not require citation information.

Keywords: Collaboration networks · Publication impact · Text mining · Semantic similarity · Semantometrics

1 Introduction

It has been shown scientific authorship is shifting from single-authored publications towards team production [23] and international collaboration [22]. Consequently, many studies have focused on scientific collaboration networks [17], patterns of scientific collaboration across disciplines [4], and how these patterns affect scientific production and impact [9]. Many such studies have focused on the concept of “bridges” – collaboration ties formed by authors from different communities or fields which create bridges between these distinct communities

or fields [8]. Within this area, it has been shown that newcomers in a group of collaborators can increase the impact of the group [9], and that high impact scientific production occurs when scientists create connections across otherwise disconnected communities from different knowledge domains [14].

Existing works studying scientific collaboration networks have often focused either on properties of the network or on topical information pertaining to the nodes in the network. In this work we develop and test an approach which combines both network and topical information about the nodes. In order to gain insight into the types of collaboration between authors, we investigate the possibility of utilizing semantic distance in co-authorship networks together with the concept of *research endogamy* [16] – the tendency to collaborate with the same authors or within a group of authors; and study how these types of collaboration reflect scientific importance.

In contrast to previous studies combining topical and network information [6, 12], our approach is beneficial in that it does not require citation information or a complete network, and can therefore be applied to newly published works. This approach, which we have introduced in a previous publication [11], belongs to a class of methods referred to as “semantometrics” [13]. In contrast to the existing metrics such as bibliometrics, altmetrics or webometrics, which are based on measuring the number of interactions in the scholarly network, semantometrics build on the premise that full-text is needed to understand scholarly publication networks and the value of publications.

The content of this paper is organized as follows. First, in Sect. 2, we discuss previous work related to our research, and our motivation for utilizing research endogamy and semantic distance of authors. In Sect. 3, we first define research endogamy and author distance and present a classification of research publications created using these two measures. Next, in Sect. 3.1 we describe our methodology and in Sect. 3.2 we describe the dataset used in our study.

2 Related Work

In this section, we review previous literature relevant to our study. First, we discuss methods for measuring the strength of ties in academic social networks, particularly research endogamy. Next, we briefly discuss methods for detecting communities in scholarly networks.

2.1 Strength of Ties in Academic Social Networks

Academic social networks represent relationships among researchers. Uncovering and studying patterns of academic social networks has been applied to many problems ranging from identifying influential researchers [5] and ranking conferences [19] to measuring research contribution [18] and the diffusion of innovation [21]. One of the first studies focusing on the strength of ties in social networks [8] introduced the concept “weak ties”, i.e. ties across rather than within different communities or groups, and discussed the importance of these ties for diffusion

processes. This has later been applied to studying academic social networks [4]. The tactic used to measure the strength of the tie between two individuals has in this case been to measure the proportion of common ties shared by the two individuals [8]. Other approaches used to measure the strength of ties have been the frequency of contact [7], mutual acknowledgement of contact [4], or the likelihood of a tie re-appearing in the future [1]. [17] has proposed a measure of closeness of two authors which combines information about how many papers two authors wrote together and the number of other collaborators with whom they wrote them.

Following the ideas of [8] and later [9], who classified agents in a network as incumbents and newcomers, and have shown newcomers to a group help to improve its performance, [16] have used the degree of new collaborations to rank conferences. The degree of new collaborations has been quantified using a new indicator called “research endogamy”, which captures the inclination of a group to usually collaborate together. [16] have shown the reputability of computer science conferences is correlated with the endogamy of their authors – low endogamy (i.e. less frequent collaboration) tends to be associated with highly reputed conferences, while lower quality conferences tend to publish articles by authors who have collaborated together on many occasions. [19] have applied the concept of endogamy to ranking publications and patents, and have shown low endogamy publications tend to receive more citations.

Overall, the aforementioned studies demonstrate the importance of connections across communities, diverse collaborations, and newcomers to a group. These patterns tend to be associated with high impact academic production. Hence, in this work, we use the concept of research endogamy of publications as defined by [19] to measure the strength of collaboration of a group of authors.

2.2 Semantic Similarity for Community Detection

Two approaches commonly used to detect communities in academic social networks are: (1) using the graph structure of the network or (2) using textual information of the nodes, e.g. by calculating semantic similarity between the nodes [3]. These two approaches have also been used together to create maps of scientific communities in a specific field [6, 12] and to identify similar researchers [2]. However, the network-based approach poses a significant challenge. Community detection in incomplete networks is a challenging task which requires the use of non-traditional methods [15]. However, the complete network may not always be available, or may be difficult to obtain. For example, in order to identify whether two authors are members of the same community or of different communities, complete information about each of their communities (other authors and links between them) are needed.

Furthermore, network-based community detection has been shown to result in communities which span diverse topics, while text-based community detection helps in detecting nodes focusing on a specific topic [3]. As we are interested in studying individual publications for which we may not have complete neighborhood information, we chose the text-based approach, and use semantic

distance (the inverse of similarity) to measure the similarity of authors. This is also beneficial, as the textual similarity provides information complementary to the endogamy measure, which is calculated using topological information. By combining these two approaches, we are able to study collaboration networks not only from the perspective of tie strength, but also from the perspective of whether each tie represents potential knowledge transfer within or across disciplines.

3 Approach and Dataset

In [10], we have proposed a classification of research publications in which publications are divided into four groups (Fig. 1) according to the semantic distance and the strength of ties between the publications’ authors. In this paper, we provide an evaluation of this approach. To do this, we use the recently released TrueImpactDataset [11] which contains publications of two types, seminal publications and literature reviews, and compare the collaboration patters of these two types of publications in terms of author distance and collaboration frequency.

The semantic distance of a pair of authors is calculated using their previous publication record. Figure 2 illustrates which publications are used in the calculation. For example, distance between authors a_1 and a_2 in Fig. 2 would be calculated using distance between the following two sets of publications: $\{p_1, p_2, p\}$ and $\{p, p_3, p_4\}$. Specifically, we measure the semantic distance $d(p)$ between authors of publication p as a mean semantic distance between all pairs of authors:

$$d(p) = \frac{1}{|A(p)| \cdot (|A(p)| - 1)} \sum_{a_i \in A(p), a_j \in A(p), a_i \neq a_j} d(a_i, a_j) \tag{1}$$

Here $A(p)$ is a set of authors of publication p . As explained in [10], we calculate the distance for a pair of authors $d(a_i, a_j)$ by concatenating the publications

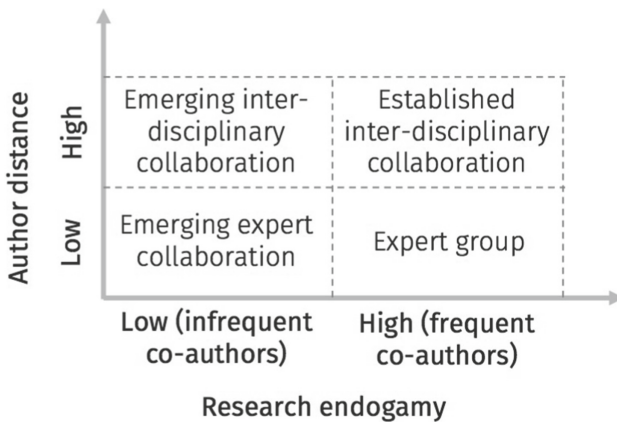


Fig. 1. Types of research collaboration based on semantic distance of authors, and their collaboration frequency.

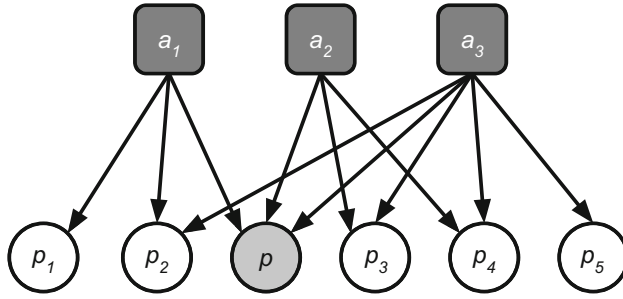


Fig. 2. A sample network showing the set of publications (round nodes) and authors (squared nodes) used in the calculation of author distance and research endogamy of publication p .

of each author into a single document. While this is a very simplistic approach, it is also beneficial in terms of complexity of the calculation.

In order to measure the strength of ties between authors, we combine the semantic distance with research endogamy value of the publication. Research endogamy [16] is the tendency to collaborate with the same authors or within a group of authors. The research endogamy of a publication is calculated based on research endogamy of a set of authors A , which is defined similarly as the Jaccard similarity coefficient [16, 19] (Eq. 2). The authors and publications used in the calculation are depicted in Fig. 2. The research endogamy $e(A)$ of a set of authors is calculated as follows:

$$e(A) = \frac{|\bigcap_{a \in A} P(a)|}{|\bigcup_{a \in A} P(a)|} \tag{2}$$

Here $P(a)$ represents a set of papers written by author a . Higher endogamy value is related to more frequent collaboration between authors in A – a value of 1 means all authors in A have written all of their publications together. On the other hand, a group of authors who have never collaborated together will have an endogamy value of 0. For example, the research endogamy of authors a_1 and a_2 in Fig. 2 is $1/5$, while the endogamy of authors a_2 and a_3 is $3/5$, i.e. authors a_2 and a_3 tend to collaborate more frequently than authors a_1 and a_2 .

Endogamy of a publication p is then defined as a mean of endogamy values of the power set of its authors [16, 19] (Eq. 3).

$$e(p) = \frac{\sum_{x \in L(p)} e(x)}{|L(p)|} \tag{3}$$

Here $L(p)$ is the set of all subsets with at least two authors of p , $L(p) = \bigcup_{k=2}^{|A(p)|} L_k(p)$, where $L_k(p) = C(A(p), k)$ is the set of all subsets of $A(p)$ of length k .

3.1 Methodology

To study the relation between author distance and research endogamy we use our TrueImpactDataset, a multidisciplinary dataset of research publications containing seminal publications and literature reviews. We are interested in how these two types of papers are situated with regard to author distance and research endogamy. However, we also look at whether the two measures relate to the number of citations each publication received. A correlation would suggest the two metrics could potentially assist in predicting the future citation counts. Finally, we compare research endogamy and author distance, and citation counts in terms of how well each method distinguishes between seminal publications and literature reviews.

We use the following methodology. For the publications in the dataset we collect and/or calculate the following measures: (1) author distance, (2) research endogamy, (3) collaboration category (assigned to publications using author distance and research endogamy, Fig. 1), (4) total number of citations per publication, (5) number of citations normalized by number of authors, and (6) number of citations normalized by publication age. To compare seminal publications and literature reviews in our dataset with regards to author distance and research endogamy we use t and χ^2 tests to determine whether the values of the measures are statistically significant for seminal publications and literature reviews. To analyze whether author distance and research endogamy help in distinguishing between seminal publications and literature reviews in our dataset we also analyze the distributions of both features and the placement of seminal publications and literature reviews within the four collaboration categories (Fig. 1).

3.2 Data

To collect all data needed for studying the measures introduced in Sect. 3, we have used three data sources:

1. TrueImpactDataset¹ [11], which provides us with seminal publications and literature reviews (i.e. the p node in Fig. 2),
2. Microsoft Academic (MA) API² [20] which we use to collect metadata (particularly the information about authors and their publications, gray and yellow nodes in Fig. 2) of the papers in the TrueImpactDataset,
3. Mendeley API³ which we use to collect publication abstracts.

Table 1 shows the size of the dataset. After collecting all needed data the size of the original dataset was reduced to 144 publications (i.e. publications for which we were able to obtain author information) – 75 literature reviews and 69 seminal publications. The rows *Total authors* and *Unique authors* show the total number of authors of all papers in the dataset and the number of unique

¹ trueimpactdataset.semantometrics.org/.

² aka.ms/academicgraph/.

³ dev.mendeley.com/.

author names, respectively. To count the unique names, we have compared the surname and all first name initials, in case of a match we consider the names to be the same (e.g. J. Adam Smith and John A. Smith will be counted as one unique name). The number of unique author names doesn't show the number of disambiguated authors, but gives us an indication of how many of the author names repeat in the dataset.

Table 1. Dataset size. The table shows for how many of the TrueImpactDataset publications we managed to get the needed metadata and how many additional publications we collected (i.e. including all other publications of the authors in the TrueImpactDataset – row *Total number of publications*).

Publications in TrueImpactDataset	314
TrueImpactDataset publications in MA	298
Pubs with author information in MA	144
Total authors	758
Unique authors	727
Total number of publications	27,653

4 Experiments

We begin by comparing the properties of survey publications and literature reviews. We investigate how these two types of papers are situated with regard to the extracted features. To do this, we use the following methodology: we take all of the 144 core papers and for each of them collect the features defined in Sect. 3.1. To understand whether seminal publications and literature reviews differ in terms of these features we calculate an independent one-tailed *t*-test for each feature except for the collaboration category feature which is categorical and for which we calculate χ^2 test. The *t*-test is a measure commonly used to assess whether two sets of data are statistically different from each other. In other words, it helps to determine the features that can distinguish survey papers from seminal papers. To test the significance, we set the significance threshold at 0.05. Furthermore, for each feature we create a histogram and by comparing these histograms for the two publication types we gain insight into norms and placement of seminal and survey publications in terms of metrics.

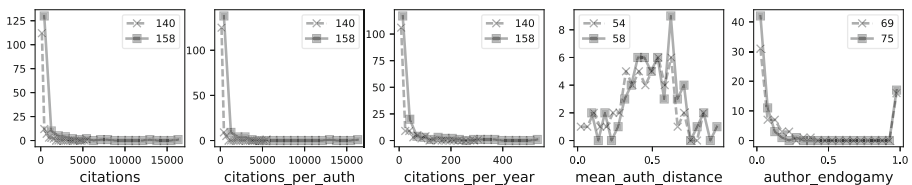


Fig. 3. Histograms of the five numerical features.

The complete results of the t -test are presented in Table 2 and the histograms for the five numerical features are shown in Fig. 3. For four of the features we reject the null hypothesis of equal means. The t -test tells us the values of these four features are significantly different for the two sets of papers.

Table 2. Results of t - and χ^2 tests.

Metric	p -value
Mean author distance	0.0327
Endogamy	0.3217
Citations	0.0012
Citations per year	0.0073
Citations per author	0.0110
Collaboration category	0.0218

Next, we analyze the collaboration category feature which is assigned to publications using the values of author distance and research endogamy (Fig. 1). We calculate χ^2 test, which is a statistical test for categorical variables for testing whether the means of two groups are the same, to test whether the seminal publications and literature reviews differ in terms of the collaboration category. The resulting p -value is 0.0218 (Table 2), which is lower than our significance threshold of 0.05. This tells us that the means of the two sets of papers differ.

The relation between author distance and research endogamy is shown in Fig. 4. The labels in the figure correspond to the four collaboration categories presented in Fig. 1. Each point in the figure represents one publication, with seminal publications and literature reviews distinguished by color. The horizontal and vertical lines in the figure represent median values for each axis – the vertical line represents median endogamy value (0.0297) and the horizontal line represents median author distance value (0.4996). The median values were used to separate the publications into the four categories (Fig. 1). Figures 4 and 3 show the endogamy values for the dataset are strongly skewed towards 0. Furthermore, the results of the t -test suggest research endogamy by itself does not distinguish between the two publication types. However, when combined with the author distance measure, a clear pattern emerges. This is visible in Fig. 5, which shows number of publications of each types belonging to each collaboration category.

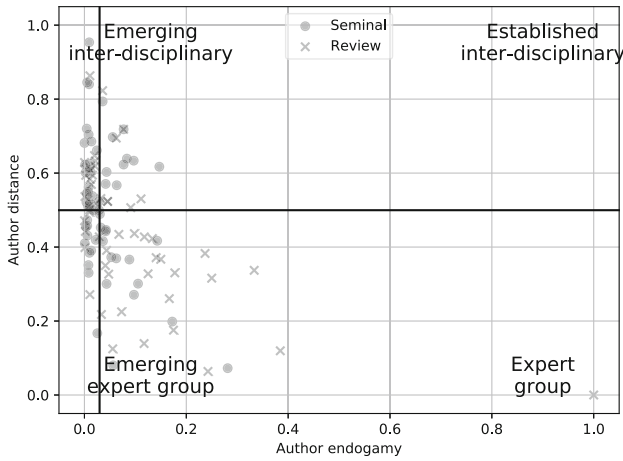


Fig. 4. Distribution of publications according to author distance and author endogamy. The horizontal and vertical lines are used to separate the publications into the four quadrants presented in Fig. 1.

Figure 5 shows there are some differences between seminal publications and literature reviews. In particular, the main difference between the two classes is that emerging collaborations (i.e. when the authors have not collaborated frequently together previously) are in our dataset more common for seminal publications. On the other hand, literature reviews seem to be a result of established collaborations within a discipline. These observations are consistent with previous studies which have shown that cross-community citation and collabora-

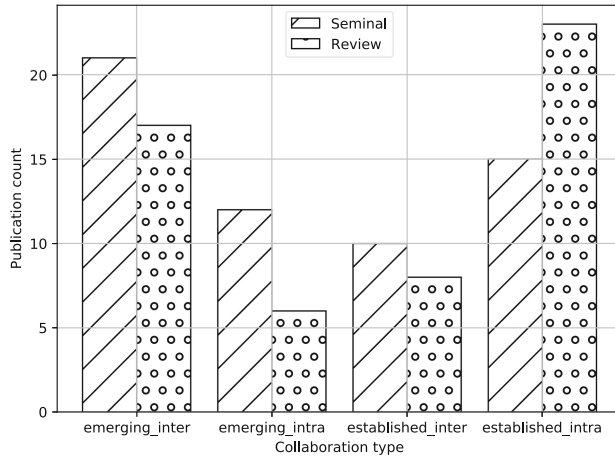


Fig. 5. Number of publications belonging to each collaboration category across both publication types.

tion patters are characteristic for high impact scientific production [9, 14, 16]. We believe this is an encouraging result which suggest semantic distance of authors combined with their endogamy value might be helpful in providing early indication of future impacts of a publication.

5 Conclusions

This paper studied the relationship between semantic distance of authors which collaborated on a publication and the strength of ties between these authors, which was assess using research endogamy measure (a measure of collaboration frequency introduced by [16]). More specifically, we compared publications of two types – seminal publications and literature reviews – in terms of their author distance and research endogamy values. Our results show that there are distinct differences between these two publication types in terms of collaboration patters. In particular, we found that seminal publications tend to be written by authors who have previously worked on dissimilar problems (i.e. authors from different fields or even disciplines), and by authors who are not frequent collaborators (i.e. emerging inter-disciplinary collaborations). On the other hand, literature reviews in our dataset tend to be the result of an established collaboration within a discipline (an “expert group”). This demonstrates content analysis might provide valuable information for research evaluation and meaningful information about potential future impacts of a publication which does not require citation information.

Acknowledgments. This research was supported in part by an appointment to the Oak Ridge National Laboratory ASTRO Program, sponsored by the U.S. Department of Energy and administered by the Oak Ridge Institute for Science and Education.

This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

This manuscript has been authored by UT-Battelle, LLC under Contract No. DE-AC05-00OR22725 with the U.S. Department of Energy. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (<http://energy.gov/downloads/doe-public-access-plan>).

References

1. Brandao, M.A., Vaz de Melo, P.O.S., Moro, M.M.: Tie strength persistence and transformation. In: AMW (2017)
2. Cabanac, G.: Accuracy of inter-researcher similarity measures based on topical and social clues. *Scientometrics* **87**(3), 597–620 (2011)
3. Ding, Y.: Community detection: topological vs. topical. *J. Informetr.* **5**(4), 498–514 (2011)

4. Friedkin, N.: A test of structural features of granovetter's strength of weak ties theory. *Soc. Netw.* **2**(4), 411–422 (1980)
5. Fu, T.Z.J., Song, Q., Chiu, D.M.: The academic social network. *Scientometrics* **101**(1), 203–239 (2014)
6. Glenisson, P., Glänzel, W., Janssens, F., De Moor, B.: Combining full text and bibliometric information in mapping scientific disciplines. *Inf. Process. Manag.* **41**(6), 1548–1572 (2005)
7. Granovetter, M.: The strength of weak ties: a network theory revisited. In: *Sociological Theory*, pp. 201–233 (1983)
8. Granovetter, M.S.: The strength of weak ties. *Am. J. Sociol.* **78**, 1360–1380 (1973)
9. Guimerà, R., Uzzi, B., Spiro, J., Amaral, L.A.N.: Team assembly mechanisms determine collaboration network structure and team performance. *Science* **308**, 697–702 (2005)
10. Herrmannova, D., Knoth, P.: Semantometrics in coauthorship networks: fulltext-based approach for analysing patterns of research collaboration. *D-Lib Mag.* **21**(11/12) (2015)
11. Herrmannova, D., Patton, R.M., Knoth, P., Stahl, C.G.: Citations and readership are poor indicators of research excellence: introducing trueimpactdataset, a new dataset for validating research evaluation metrics. In: *Proceedings of the 1st Workshop on Scholarly Web Mining* (2017)
12. Janssens, F., Leta, J., Glänzel, W., De Moor, B.: Towards mapping library and information science. *Inf. Process. Manag.* **42**(6), 1614–1642 (2006)
13. Knoth, P., Herrmannova, D.: Towards semantometrics: a new semantic similarity based measure for assessing a research publication's contribution. *D-Lib Mag.* **20**(11), 8 (2014)
14. Lambiotte, R., Panzarasa, P.: Communities, knowledge creation, and information diffusion. *J. Informet.* **3**(3), 180–190 (2009)
15. Lin, W., Kong, X., Yu, P.S., Wu, Q., Jia, Y., Li, C.: Community detection in incomplete information networks. In: *Proceedings of the 21st International Conference on World Wide Web*, pp. 341–350. ACM (2012)
16. Montolio, S.L., Dominguez-Sal, D., Larriba-Pey, J.L.: Research endogamy as an indicator of conference quality. *SIGMOD Rec.* **42**(2), 11–16 (2013)
17. Newman, M.E.J.: Who is the best connected scientist? A study of scientific coauthorship networks. In: Ben-Naim, E., Frauenfelder, H., Toroczkai, Z. (eds.) *Complex Networks*, pp. 337–370. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-44485-5_16
18. Rocha, L., Moro, M.M.: Research contribution as a measure of influence. In: *Proceedings of the 2016 International Conference on Management of Data*, pp. 2259–2260. ACM (2016)
19. Silva, T.H.P., Moro, M.M., Silva, A.P.C., Meira Jr., W., Laender, A.H.F.: Community-based endogamy as an influence indicator. In: *Digital Libraries 2014 Proceedings*, p. 10 (2014)
20. Sinha, A., et al.: An overview of Microsoft academic service (MAS) and applications. In: *Proceedings of the 24th International Conference on World Wide Web*, pp. 243–246. ACM (2015)
21. Valente, T.W.: Social network thresholds in the diffusion of innovations. *Soc. Netw.* **18**(1), 69–89 (1996)
22. Witze, A.: Research gets increasingly international. *Nature* **785**, 6–8 (2016)
23. Wuchty, S., Jones, B.F., Uzzi, B.: The increasing dominance of teams in production of knowledge. *Science* **316**(5827), 1036–1039 (2007)



Aspect Specific Opinion Expression Extraction Using Attention Based LSTM-CRF Network

Abhishek Laddha¹ and Arjun Mukherjee²(✉)

¹ Indian Institute of Technology Delhi, New Delhi 110016, India

² Department of Computer Science, University of Houston, Houston, TX, USA
arjun@cs.uh.edu

Abstract. Opinion phrase extraction is one of the key tasks in fine-grained sentiment analysis. While opinion expressions could be generic subjective expressions, aspect specific opinion expressions contain both the aspect as well as the opinion expression within the original sentence context. In this work, we formulate the task as an instance of token-level sequence labeling. When multiple aspects are present in a sentence, detection of opinion phrase boundary becomes difficult and label of each word depend not only upon the surrounding words but also with the concerned aspect. We propose a neural network architecture with bidirectional LSTM (Bi-LSTM) and a novel attention mechanism. Bi-LSTM layer learns the various sequential pattern among the words without requiring any hand-crafted features. The attention mechanism captures the importance of context words on a particular aspect opinion expression when multiple aspects are present in a sentence via location and content based memory. A Conditional Random Field (CRF) model is incorporated in the final layer to explicitly model the dependencies among the output labels. Experimental results on Hotel dataset from Tripadvisor.com showed that our approach outperformed several state-of-the-art baselines.

Keywords: Sentiment analysis · Opinion expressions · Conditional random field

1 Introduction

Aspect based sentiment analysis [15] is one of the main frameworks for fine-grained sentiment analysis and is used in several downstream tasks such as opinion summarization, extracting opinion targets, opinion holders, opinion expressions etc. One of the main goals of aspect based sentiment analysis is to identify the fine-grained product properties (aspects) and their opinion. In [10, 16, 24] the aspect term and opinion words are jointly extracted but lack correspondence between the aspect and opinion terms. For example, in the sentence “*the food was excellent and plentiful* and the *waitstaff was extremely friendly and helpful*”,

discovering aspect words as $\{food, waitstaff\}$ and opinion words as $\{excellent, plentiful\}$ etc.} is definitely useful but being able to extract phrases that retain the sentence context as aspect specific opinion expressions such as (*food was excellent and plentiful, waitstaff was extremely friendly and helpful*) would be more expressive and provide more information about the aspect. These opinion phrases can be further used in downstream application such as aspect sentiment classification, aspect summarization.

Traditionally, subjective expression extraction [2, 3] has been formulated as a token-level sequence labeling task and has employed a CRF based approach using hand-crafted features. Recent success of distributed representation of words [18, 20] provides alternate approach to learn the continuous valued dense vectors for latent features in hidden layers. [9, 16] apply deep Recurrent Neural Network (RNN) to extract the opinion expression and opinion target from sentences. They have shown that a deep RNN approach outperforms traditional CRF and semi-CRF. Approaches in [9, 16] learn the opinion phrase representation based on the latent features learned of context words but are incapable of explicitly providing the cues from the aspect word. [22, 23, 28] proposed models to extract the sentiment of an aspect in a sentence by taking into account the aspect words. They are mainly focusing on positive or negative sentiment instead of generic opinion phrase about aspect.

In this paper, we present a neural network architecture with Bi-LSTM and an attention mechanism to take into account the aspect cues. Bi-LSTM layer learns the various sequential patterns among the words without requiring any hand-crafted features. Most of the current work in aspect classification [22, 23, 28] assumes presence of one aspect in the sentence. If there are multiple aspects in the same sentence they consider them as separate instance ignoring the effect of one aspect on another. We believe that if there are multiple aspects in a sentence, explicitly feeding the importance of context word based on the content and location for a particular aspect is an essential signal to decide whether a context word is in opinion expression of aspect or not.

Inspired by recent success of attention based computational model as in aspect sentiment classification [23, 28], machine translation [1], we propose an attention mechanism which takes into account the multiple aspects in the sentence based on the context/surrounding word's location from multiple aspect word. This layer would be helpful in tagging the words which are in between the two aspect and can be included into both aspect opinion expression thereby helping in locating the precise boundary of aspect specific opinion phrases. A CRF model is incorporated in the final layer to explicitly model the dependencies among the output labels.

2 Related Work

We briefly review the existing studies on subjective expression extraction task and aspect-based sentiment analysis using neural networks in this section.

2.1 Subjective Expression Extraction

Early works on fine-grained opinion extraction [2, 3] have used various parsing, syntactic, lexical and dictionary based features to extract a subjective expression employing a CRF based approach. Various features based on dependency relations [11] and opinion lexicon have been used for opinion expression extraction. Further, [29, 30] employed semi-CRFs which allowed sequence labeling at the segment level. [30] proposed a joint inference model to jointly detect opinion expressions, opinion holders and target as well as relation among them. While they have made important progresses, their performances mainly rely on rich hand crafted features and other pre-processing steps such as dependency parsing.

There have been work exploring the combinations of sequential neural network (e.g. LSTM, RNN) on sequence labeling tasks such as Named Entity Recognition (NER), language understanding. [8, 14, 31] added a CRF layer on top of RNN network and showed performance improvement on Named Entity Recognition (NER) and language understanding. [17] extended above model by using CNN on character of words to get word level representation. These works have mostly explored the neural networks in NER as opposed to opinion phrase extraction.

The works in [12, 26], are the closest to ours as they focus on aspect specific opinion terms. While [26] does not discover phrases, [12] employs higher order CRF features for phrase extraction and is considered as a baseline.

2.2 Neural Network for Aspect Based Sentiment Analysis

Recent studies have shown that deep learning models can automatically learn the inherent semantics and syntactic information from data and this achieves better performance for sentiment analysis. Regarding aspect based sentiment analysis [16, 27, 32] models target aspect term extraction as a sequence tagging task using neural network. In [16], RNN and word embedding were combined to extract explicit aspects. In [27], recursive neural network based on dependency tree and CRF were integrated in a unified framework to extract the aspect and opinion terms. [32] used word and dependency paths embeddings as features in CRF. These methods are mostly focused on aspect term extraction instead of aspect specific opinion expression.

Also related are the works around aspect based sentiment classification [22, 23, 28] and the work in [19] which proposed an extension of RNN to identify the aspect's sentiment. [28] proposed an LSTM model with attention mechanism which focuses on different part of sentences given the aspect input. Further, in [23], a deep memory network was proposed for explicitly capturing the importance of each context word when inferring the polarity of given aspect. These approaches provide the sentiment about the aspect but do not give in-depth information about the aspect such as the associated opinion expression.

3 Background: LSTM-CRF Model

This section briefly describes bidirectional LSTM-CRFs that lays the foundation for the proposed attention based LSTM-CRF network. For more details, refer to [8, 14, 17].

3.1 Bidirectional LSTM Network

Recurrent Neural Networks (RNNs) are a family of neural networks that take an input $X = \{x_1, \dots, x_T\}$ and yield the sequence hidden representation $\{h_1, \dots, h_T\}$ where each $h_t \in \mathbb{R}^{d \times 1}$ represents the semantic information of the left context of x_t . In practice these models fail to capture the long term dependencies and suffer from the vanishing gradient problem. [6] proposed an LSTM cell which eliminates these problems by employing several gates to control the information flow in the cell. For each word of a given sentence, an LSTM computes a representation \vec{h}_t of the left context of sentence. Similarly, a right context information \overleftarrow{h}_t also contains the useful information. This can be achieved by employing another LSTM network which reads the current sentence in the reverse order. Now, the word representation in this bidirectional LSTM would be a concatenation of its left and right latent context representations $h_t = [\vec{h}_t; \overleftarrow{h}_t] \in \mathbb{R}^{2d \times 1}$.

3.2 LSTM-CRF Model

For the task of sequence tagging, a very simple approach would be to predict the labels y_t independently for each token using a simple feed-forward neural network classification model which takes the output of the LSTM as input vectors. But for labeling of opinion expressions, there is a strong dependency associated with the previous and current label. Therefore, instead of predicting label for each token independently, we model them jointly using CRF [13]. Let's consider $P \in \mathbb{R}^{T \times q}$ to be a matrix of the output of a Bi-LSTM after projecting it on to a linear layer whose size is equal to number of distinct labels q , T is the number of tokens. We also define a transition matrix $A \in \mathbb{R}^{q \times q}$ where each entry $a_{i,j}$ represents a probability of transition from state i to j of consecutive output labels. Then, score for a complete sentence can be defined as follows

$$s(X, y) = \sum_{t=0}^T A_{y_t, y_{t+1}} + \sum_{t=1}^T P_{t, y_t} \quad (1)$$

where start state and end state is added in the sentence. Since we are considering only bigram interactions among labels, dynamic programming [13] can be used to compute A and the best possible sequence labels during inference [4].

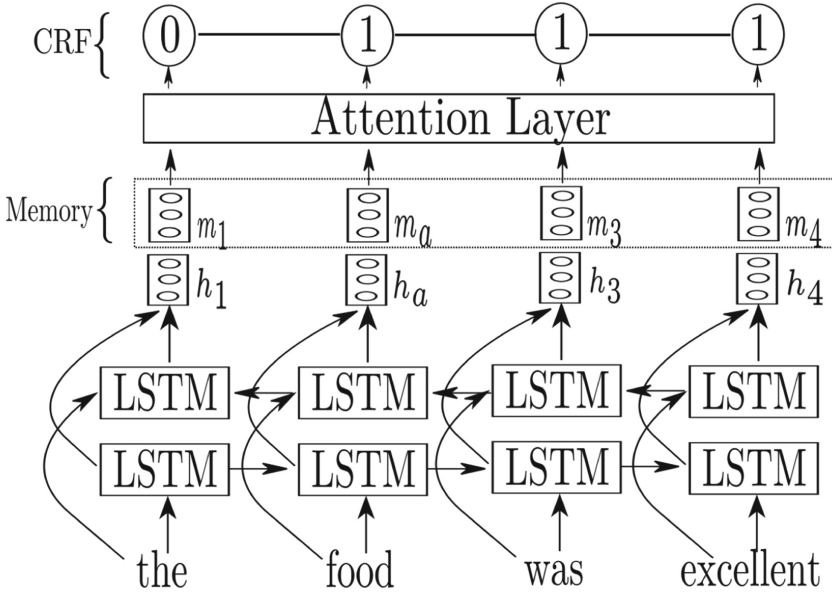


Fig. 1. Main Architecture of network. Word input are given as word embedding to Bi-LSTM.

4 Attention Based LSTM-CRF Network (LSTM-ATT-CRF)

4.1 Task Definition and Notation

Given a sentence $X = (w_1, \dots, w_T)$ with T number of words and set of n aspect words $\{w_{a_1}, \dots, w_{a_n}\}$ where $a_k \in [1, T]$ is mentioned in sentence X . Our task is to extract set of relevant opinion expression containing the aspect and opinion phrase. We formulated this as sequence labeling problem where each word of a sentence has label $y_t \in \{0, 1\}$. y_t is assigned to 1 if a word is in any aspect specific opinion expression or 0 otherwise. To represent each word, we map it onto a low dimensional continuous vector and corresponding to each word w_t , it's embedding is represented as $v_t \in \mathbb{R}^{d_w \times 1}$ where d_w is the word embedding size. The complete architecture of network is shown in Fig. 1 where Bi-Directional LSTM is similar to network described in the previous section.

4.2 Attention Network

The basic idea behind attention is to assign importance to each context word based on the latent representation and memory. In our setting, memory refers to the multi aspect information present in a sentence. Using the location of the context word from the multiple aspects, memory vector m_t is computed using Eq. 2. Main intuition is that every aspect doesn't contribute equally to determine

	food	was	excellent	and	plentiful	and	the	waitstaff	was	extremely	friendly
Before attention:											
$P(y=1)$	0.6	0.43	0.79	0.64	0.46	0.3	0.2	0.6	0.45	0.76	0.82
$P(y=0)$	0.4	0.57	0.21	0.36	0.54	0.7	0.8	0.4	0.55	0.24	0.18
After attention:											
$P(y=1)$	0.7	0.45	0.86	0.64	0.56	0.3	0.2	0.7	0.47	0.83	0.9
$P(y=0)$	0.3	0.55	0.14	0.36	0.44	0.7	0.8	0.3	0.53	0.17	0.1

Fig. 2. An illustration of example of our neural attention network for aspect specific opinion labeling. Words in blue and red corresponds to opinion expression about “food”, and “waitstaff” respectively. (Color figure online)

the polarity of each context word in the sentence. The words which are distant from the particular aspect word would influence less by that aspect word. In this work, we define the location of context words as it’s absolute distance with the aspect in the original sentence. Memory vector for each token at t is defined

$$m_t = \frac{\sum_{k=1}^n v_{a_k} \left(1 - \frac{l_{t,k}}{T}\right)}{n} \quad (2)$$

$$g_t = \tanh(W_{attn} [m_t; h_t] + b_{attn}) \quad (3)$$

where $l_{t,k}$ is number of word between w_t and w_{a_k} , $W_{attn} \in \mathbb{R}^{(2d+d_w) \times 1}$, v_{a_k} is the embedding vector of aspect a_k . Based on the memory vector and hidden representation, the model assigns an score g_t to each context word using Eq. 3 which takes into account the relation between word and multiple aspects. After getting the g_t ’s we feed them to softmax function to get the importance weights α_t for each context word, such that $\sum_t \alpha_t = 1$ and each $\alpha_t \in [0, 1]$.

$$P_t = W_{linear}^{Tr} (\alpha_t \times h_t) + b_{linear} \quad (4)$$

A linear layer is applied to transform the hidden representation vector to the scores of each output tag using the Eq. 4, where $W_{linear} \in \mathbb{R}^{2d \times q}$ and $b_{linear} \in \mathbb{R}^{q \times 1}$ and after that score of a sequence was calculated using Eq. 1. Here, P_{t,y_t} is unscaled probability of word w_t having label y_t . In the absence of attention α_t will become 1 however, attention will provide weighted hidden state with respect to aspect words. Figure 2 shows a example where word “plentiful” have low confidence in inclusion of opinion expression due to long distance from aspect word “food” and closeness to aspect word “waitstaff” for which it doesn’t express opinion. While attention will learn to give more importance to those words because there is direct interaction of hidden vector with aspect word and there are lots of opinion expression about aspect “food” which includes the words of similar meaning as of “plentiful”.

4.3 Model Training

The model can be trained end-to-end using backpropogation, where the objective function is to maximize the log-probability of correct sequence score as defined in Eq. 5.

$$p(y|X) = \frac{\exp(s(X, y))}{\sum_{\hat{y}} \exp(s(X, \hat{y}))} \quad (5)$$

where X denotes the sequence of words and y is the corresponding sequence label. $s(X, y)$ is score defined in Eq. 1, P_t learns the probability of each label independently from Bi-LSTM while A learns the dependency among the various labels. For *e.g.*, in Fig. 2 some stopwords such as “was” could have low probability of label $y_t = 1$, but decoding complete sequence using Eq. 1 will consider surrounding label and hence inclusion of such words in opinion expression. Model parameters are the LSTM weights, $W_{attn}, W_{linear}, b_{attn}, b_{linear}, A$ and word embeddings. Except word embeddings, other parameters are initialized using sampling from uniform distribution $U(-\epsilon, \epsilon)$, where $\epsilon = 0.01$.

Word Embeddings: Word embeddings are initialized using the pre-trained embeddings. We use Stanford’s publicly available GloVe¹ 100-dimensional embeddings [20]. We also experimented with two other embeddings, namely Senna² 50-dimensional embeddings [4] and Google’s word2vec³ 300-dimensional [18]. Parameter d_w depends on dimension of pre-trained word embedding vectors.

Features: Although NNs learn the word features (i.e. embedding) automatically, [16] shows that incorporating other linguistic features like part of speech (POS) and syntactic information (e.g., phrase chunks) helps in the training to learn a better model. We used the syntactic features (POS tags) and phrase chunk features as input in the LSTM network. Similar to word embedding, each feature is also mapped to feature embedding which gets learned during training. Input to LSTM network is a concatenation of word embedding and feature embeddings.

5 Experiments

5.1 Dataset

To demonstrate the effectiveness of our model we performed experiments on the Hotel dataset from Tripadvisor.com used in [25]. Labeling opinion phrases for each aspect is a tedious task and require lots of human effort. Further, training deep learning model generally needed substantial amount of training data. To overcome this bottleneck, we tried to label the phrases using heuristic rules. Seed words for each aspect used in [25] were used to label the location of aspect words in review sentences. Once, aspect words in sentence were determined, opinion expressions around those aspect words are labeled as described next.

Labeling Using Heuristic Rules. Since, we mainly focus on opinion expressions surrounding the aspect word, heuristic rules could be written with the help of Part of Speech (POS) tags and polarity of the surrounding words. We used

¹ <http://nlp.stanford.edu/projects/glove/>.

² <http://ronan.collobert.com/senna/>.

³ <https://code.google.com/archive/p/word2vec/>.

the opinion lexicon⁴ derived from [7] for positive and negative words. Labeling of the phrase boundary surrounding the aspect word for positive opinions phrases was done as follows:

In the first step, we searched for the positive terms (using the sentiment lexicon) in the window of $[-5, 5]$ around the aspect word. To have a compact opinion phrase, it should not include the opinion about the other aspects. So, only considering the extremes with a sequence index of [aspect, positive-term] would not yield good phrases. Therefore, we divided the presence of positive words in two cases. First, when the positive word was before the aspect word, to capture all the opinion words in a phrase talking about noun aspect, we took the farthest adjective from the aspect word. If aspect word was verb, then we took the nearest adverb since adverbs are mostly situated immediately before the verb. Second, when the positive word is after the aspect word, we took the nearest adjective (from the aspect word) because adjectives are generally immediately followed by nouns. We also included the negative phrases with negator words in surrounding window by finding negative terms in window of $[-5, 5]$ and then finding negator terms such as “not”, “don’t” and following the same procedure described above. This process generally yielded us the minimum boundaries of phrases but could have omitted some opinion words which we included using the method described below.

Next, we extended the phrase boundaries using the basic definition of adjective and adverb: (i) If first word of a phrase is coming before the aspect word and it is a verb, then we look at the word before the verb, if the word was an adverb, then we include it in that phrase, (ii) If the last word of the phrase was an adjective and the next word after that was noun, then we included all the consecutive nouns after that, (iii) If last word of the phrase was an adverb, then we included the next word if it was a verb, (iv) If last word of the phrase was noun, then we included all the consecutive nouns after that. Similar process was applied for extracting negative opinion expressions as well. We explain our procedure using the following *e.g.*, “The room provided a nice view of the lagoon”, aspect word is “room” which is noun and opinion word is “nice” which is adjective. Since adjective word is after the aspect word we took the nearest adjective and extracted phrase would be “room provided a nice” which is not complete. Now, we extend this using the rule (ii) which will include the noun word “view” since it immediately follows the adjective. Thus this completes the opinion expression. We observed that following these heuristic rule are able to capture various fluid opinion expression like “wonderful hotel at a reasonable price,” and “rooms do feel quite bland”.

Dataset Dissection. Using the above procedure, we labeled a total of 10,775 sentences which was split in 80 : 20 ratio for training and validation. We want to evaluate on the dataset which is completely realistic and wanted to test the ability of our model to retrieve phrases which might not have labeled using the heuristic rules. Hence, for testing, we manually labeled another disjoint set of

⁴ <http://www.cs.uic.edu/~liub/FBS/opinion-lexicon-English.rar>.

Table 1. Comparison of results with baselines

Model	Precision	Recall	F-score
CRF	82.77	69.01	75.26
Semi-CRF	84.63	78.27	81.29
LSTM-CNN-CRF	88.46	72.47	79.67
LSTM-ATTN-CRF	88.80	75.86	81.82

1,683 sentences after getting the location of aspect labeled from the seed words. Dataset will be released to serve as language resource. We preprocessed the data by lowercasing all the word and replaced all cardinal numbers with “NUM” symbol and removed words appearing only once.

5.2 Parameters Settings

Our model was implemented in tensorflow⁵ using the Adam optimizer with initial learning rate of .01 and early stopping criteria [5] was used based on validation set accuracy. The decaying learning rate for Adam was set to 0.05. Care was taken to reduce overfitting by adding a dropout layer regularizer [21] with keep probability of 0.5 and gradients were clipped at 5. Other hyperparamters such as dimension of the hidden states of LSTM were kept same for all model $d = 100$, # of layers as 3, batch size as 10, and maximum length of sentence was set to 50 which were determined using pilot experiments.

5.3 Comparison with Baselines

We compared our model with the following most relevant baselines.

CRF: We use linear chain-CRF [13] and higher order features as described in [12].

Semi-CRF: This is due to model in [29] that used dependency tree features with semi-CRF to label the sequence at segment level.

LSTM-CNN-CRF: This is due to model in [17] which used the word and character level representation using combination of LSTM, CNN and CRF for sequence labeling task.

LSTM-ATT-CRF: Our complete proposed model which have attention over the output of Bi-LSTM using aspect memory with CRF layer.

Next, we also explored two simplified versions of our model.

LSTM-ATT: This model employs the cross-entropy between the predicted and target labels for loss instead of maximizing the CRF score.

LSTM-CRF: The concatenated output vectors of Bi-LSTM are passed directly into the linear layer for computing the CRF score.

⁵ <http://tensorflow.org/>.

5.4 Discussion

We used word based micro precision, recall and F-score to evaluate the quality of the model. [30] has shown that it is difficult to get exact opinion expression boundaries even for human annotators and hence focused on precision, recall measures at word level instead of complete expression level. Precision is defined as $\frac{|C \cap P|}{P}$ and recall as $\frac{|C \cap P|}{C}$, where C and P are the correct and predicted set of word labels respectively.

Table 2. Performance on aspect specific opinion expression task on Precision, Recall, F1 for different models when initialized using various pre-trained embeddings

Model	Senna			word2vec			Glove		
	P	R	F	P	R	F	P	R	F
LSTM-CNN-CRF	88.46	72.47	79.67	89.93	71.18	79.46	87.35	73.15	79.62
LSTM-ATT-CRF	88.80	75.86	81.82	88.40	75.08	81.20	87.73	76.30	81.62
LSTM-ATT	87.3	73.31	79.58	88.22	74.57	80.82	87.4	74.84	80.67
LSTM-CRF	88.14	75.94	81.59	88.68	74.13	80.75	87.98	75.40	81.2

Table 3. Example of attention weight for different example. Underlined words are aspect words, weights colored in blue are probably correct while weights in red are wrong. True opinion expression are included in [[]]. Higher weights mean the more probable a word is in opinion expression.

hotel	in	[[excellent	<u>location</u>	close	to	everything]]	we	were	impressed	[[excellent	<u>service</u>	at	reception	upon	arrival]]	
0.021	0.013	0.048	0.066	0.075	0.026	0.075	0.015	0.015	0.067	0.076	0.073	0.06	0.050	0.031	0.026					
there	are	[[waterfalls	in	<u>lobby</u>	area]]	and	[[free	easy	fast	<u>internet</u>	<u>access</u>]]	but	only	in	<u>lobby</u>	area
0.027	0.015	0.0187	0.024	0.107	0.066	0.083	0.103	0.106	0.103	0.04	0.026	0.038	0.019	0.016	0.027	0.016				

Table 1 illustrates the comparison results of baselines with our best model LSTM-ATT-CRF. Our model significantly outperforms CRF and LSTM-CNN-CRF on F-score. It also improves over semi-CRF at $p < 0.05$ semi-CRF performs close to our model due to the fact that many opinion phrases are noun phrases (NPs) and verb phrases (VPs), and its use of segment level labeling greatly improved recall but it suffers in precision.

Further from Table 2, we note LSTM-ATTN-CRF outperforms character based LSTM-CNN-CRF and LSTM-CRF across all word embedding which shows including the aspect information using the attention is effective even when features based on POS and syntactic tags (phrase chunk units) are included as input. Our complete model outperform LSTM-ATT significantly which shows that adding the CRF layer to capture the dependency among output label is useful. Also interesting to note is that Senna embeddings perform best for aspect specific opinion expression. This is due to the fact that Senna was trained on various NLP task such as NER, POS and SRL whereas other Glove, word2vec were trained on general co-occurrence of words.

For a yet deeper understanding of the attention mechanism, Table 3 shows the attention weights for two examples. Weights of context words around aspect confirms that our attention mechanism is able to assign the weights based on the word and aspect. Reason for incorrect word such as ‘waterfall’ is due to their low occurrences in the corpus while others are stopwords which sometimes get included in the opinion expression. Our model is able to assign the substantial weight to many neutral words such as ‘close’ and ‘everything’ based on the aspect which contributes to its effectiveness over other baselines.

6 Conclusion

In this paper, we presented an attention based LSTM-CRF (LSTM-ATT-CRF) for aspect specific opinion expression task. The model works for both single and multiple aspect sentences and improves phrase discovery by leveraging the latent interactions among the aspect and opinion words based on the content and location which we modeled via attention mechanism. Experimental results on a hotel dataset showed superior performance over several baselines. The work also produced a labeled dataset which shall be released as a resource.

Acknowledgement. This work is supported in part by NSF 1527364.

References

1. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint [arXiv:1409.0473](https://arxiv.org/abs/1409.0473) (2014)
2. Breck, E., Choi, Y., Cardie, C.: Identifying expressions of opinion in context. In: IJCAI, vol. 7, pp. 2683–2688 (2007)
3. Choi, Y., Cardie, C., Riloff, E., Patwardhan, S.: Identifying sources of opinions with conditional random fields and extraction patterns. In: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing, pp. 355–362. Association for Computational Linguistics (2005)
4. Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P.: Natural language processing (almost) from scratch. *J. Mach. Learn. Res.* **12**(Aug), 2493–2537 (2011)
5. Graves, A., Mohamed, A.R., Hinton, G.: Speech recognition with deep recurrent neural networks. In: 2013 IEEE International Conference on Acoustics, speech and signal processing (ICASSP), pp. 6645–6649. IEEE (2013)
6. Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural Comput.* **9**(8), 1735–1780 (1997)
7. Hu, M., Liu, B.: Mining and summarizing customer reviews. In: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 168–177. ACM (2004)
8. Huang, Z., Xu, W., Yu, K.: Bidirectional lstm-crf models for sequence tagging. arXiv preprint [arXiv:1508.01991](https://arxiv.org/abs/1508.01991) (2015)
9. Irsoy, O., Cardie, C.: Opinion mining with deep recurrent neural networks. In: EMNLP, pp. 720–728 (2014)

10. Jo, Y., Oh, A.H.: Aspect and sentiment unification model for online review analysis. In: Proceedings of the Fourth ACM International Conference on Web Search and Data Mining, pp. 815–824. ACM (2011)
11. Johansson, R., Moschitti, A.: Extracting opinion expressions and their polarities: exploration of pipelines and joint models. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2, pp. 101–106. Association for Computational Linguistics (2011)
12. Laddha, A., Mukherjee, A.: Extracting aspect specific opinion expressions. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp. 627–637 (2016)
13. Lafferty, J., McCallum, A., Pereira, F., et al.: Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: Proceedings of the Eighteenth International Conference on Machine Learning, ICML, vol. 1, pp. 282–289 (2001)
14. Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., Dyer, C.: Neural architectures for named entity recognition. arXiv preprint [arXiv:1603.01360](https://arxiv.org/abs/1603.01360) (2016)
15. Liu, B.: Sentiment analysis and opinion mining. Synth. Lect. Hum. Lang. Technol. **5**(1), 1–167 (2012)
16. Liu, P., Joty, S.R., Meng, H.M.: Fine-grained opinion mining with recurrent neural networks and word embeddings. In: EMNLP, pp. 1433–1443 (2015)
17. Ma, X., Hovy, E.: End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Berlin, Germany, pp. 1064–1074. Association for Computational Linguistics, August 2016. <http://www.aclweb.org/anthology/P16-1101>
18. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013)
19. Nguyen, T.H., Shirai, K.: Phrasernn: phrase recursive neural network for aspect-based sentiment analysis. In: EMNLP, pp. 2509–2514 (2015)
20. Pennington, J., Socher, R., Manning, C.D.: Glove: global vectors for word representation. In: EMNLP, vol. 14, pp. 1532–1543 (2014)
21. Srivastava, N., Hinton, G.E., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)
22. Tang, D., Qin, B., Feng, X., Liu, T.: Effective LSTMS for target-dependent sentiment classification. arXiv preprint [arXiv:1512.01100](https://arxiv.org/abs/1512.01100) (2015)
23. Tang, D., Qin, B., Liu, T.: Aspect level sentiment classification with deep memory network. arXiv preprint [arXiv:1605.08900](https://arxiv.org/abs/1605.08900) (2016)
24. Titov, I., McDonald, R.: Modeling online reviews with multi-grain topic models. In: Proceedings of the 17th International Conference on World Wide Web, pp. 111–120. ACM (2008)
25. Wang, H., Lu, Y., Zhai, C.: Latent aspect rating analysis on review text data: a rating regression approach. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 783–792. ACM (2010)
26. Wang, S., Chen, Z., Liu, B.: Mining aspect-specific opinion using a holistic lifelong topic model. In: Proceedings of the 25th International Conference on World Wide Web, pp. 167–176. International World Wide Web Conferences Steering Committee (2016)

27. Wang, W., Pan, S.J., Dahlmeier, D., Xiao, X.: Recursive neural conditional random fields for aspect-based sentiment analysis. arXiv preprint [arXiv:1603.06679](https://arxiv.org/abs/1603.06679) (2016)
28. Wang, Y., Huang, M., Zhao, L., Zhu, X.: Attention-based LSTM for aspect-level sentiment classification. In: EMNLP, pp. 606–615 (2016)
29. Yang, B., Cardie, C.: Extracting opinion expressions with semi-markov conditional random fields. In: Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pp. 1335–1345. Association for Computational Linguistics (2012)
30. Yang, B., Cardie, C.: Joint inference for fine-grained opinion extraction. In: ACL (1), pp. 1640–1649 (2013)
31. Yao, K., Peng, B., Zweig, G., Yu, D., Li, X., Gao, F.: Recurrent conditional random field for language understanding. In: 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4077–4081. IEEE (2014)
32. Yin, Y., Wei, F., Dong, L., Xu, K., Zhang, M., Zhou, M.: Unsupervised word and dependency path embeddings for aspect term extraction. arXiv preprint [arXiv:1605.07843](https://arxiv.org/abs/1605.07843) (2016)

Author Index

A

Abdennadher, Slim I-41
Achom, Amika I-299
Adriani, Mirna I-62
Afli, Haithem II-161
Alexandrov, Mikhail II-412
Alhakim Freihat, Abed I-220
Alizadeh, Pegah I-180
Allauzen, Alexandre I-349
Al-Yahya, Maha I-124
Artaud, Chloé I-193
Asghari, Habibollah I-99

B

Bagirov, Adil II-314
Bajpai, Rajiv II-142
Barinova, Ariadna I-207
Batsuren, Khuyagbaatar I-220
Belhaj Rhouma, Sourour II-400
Bellot, Patrice II-329
Ben Khelil, Cherifa II-209
Bennamoun, Mohammed II-3
Berrut, Catherine II-400
Besacier, Laurent I-349
Bhattacharyya, Pushpak I-3
Bick, Eckhard II-248
Bindlish, Ishita II-104, II-115, II-129, II-301
Brandl, Julia II-287
Bui, Van-Tan II-234
Burkell, Jacquelyn II-17
Burns, Jason II-161

C

Cambria, Erik I-87, II-142, II-169
Cardiff, John II-275, II-412
Cellier, Peggy I-180
Charnois, Thierry I-180
Chaturvedi, Iti II-169
Chelmuş, Rareş I-17
Chen, Qian II-169
Chernyak, Ekaterina I-207
Chhaya, Niyati II-71

Choudhary, Nurendra I-371, II-104, II-115,
II-129, II-301
Churkin, Elena II-261
Colhon, Mihaela II-92
Collovini, Sandra II-197
Crémilleux, Bruno I-180
Cristea, Dan I-50
Cui, Hong II-17

D

Dalyan, Tugba II-46
Das, Ranjita II-364
Debnath, Dipanwita II-364
Diac, Paul II-92
Dinu, Liviu P. I-110, I-408
Doucet, Antoine I-193
Duchier, Denys II-209
Dumitru, Bogdan I-408

E

Ekakristi, Adrianus Saga I-62
Ekbal, Asif II-221
Elmahdy, Mohamed I-41

F

Fonseca, Evandro II-197

G

Galitsky, Boris I-74
Gelbukh, Alexander I-299, II-364
Gîfu, Daniela I-17, I-50
Giunchiglia, Fausto I-220
Godard, Pierre I-349
González, José-Ángel I-131
Gorantla, Sruthi II-142
Guibon, Gaël II-329
Gupta, Deepak I-3

H

Ha, Thi-Thanh II-354
Hattori, Gen I-289
Hazarika, Devamanyu II-142

Heracleous, Panikos I-30, I-289, II-181
 Herrmannova, Drahomira II-431
 Hu, Changjian II-381
 Hurtado, Lluís-F. I-131

I

Iftene, Adrian I-17
 Ilvovsky, Dmitry I-74, I-207
 Ishikawa, Akio I-289
 Ivanov, Vladimir I-168

J

Jaidka, Kokil II-71
 Jamoussi, Salma II-80
 Jha, Jyoti II-32
 Jiang, Lili I-153

K

Kansal, Vineet I-331
 Knoth, Petr II-431
 Kohail, Sarah I-3
 Koshulko, Olexiy II-412
 Krishna Rohit, Sakala Venkata I-236
 Krishnamurthy, Gangeshwar I-87
 Kruger, Bernie II-314

L

Laddha, Abhishek II-442
 Last, Mark II-261
 Latiri, Chiraz II-400
 Lima, Thiago II-197
 Litvak, Marina II-261
 Liu, Wei II-3
 Liu, Wuying I-143
 Liu, Yanshuang II-59
 Löser, Kevin I-349

M

Mahendra, Rahmad I-62, I-416
 Majumder, Navonil I-87
 Mărănduc, Cătălina II-92
 Matsumoto, Kazunori I-289
 Mercer, Robert E. II-17
 Miao, Qingliang II-381
 Mohammad, Yasser I-30, II-181
 Mohtaj, Salar I-99
 Moraes, Sílvia II-197
 Mouromtsev, Dmitry I-207

Mukherjee, Arjun II-442
 Mundotiya, Rajesh Kumar I-382
 Murauer, Benjamin I-320, II-287

N

Naskar, Sudip Kumar II-221
 Nasri, Jihen II-80
 Nguyen, Kiem-Hieu II-354
 Nguyen, Le-Minh II-341
 Nguyen, Minh-Tien II-341
 Nguyen, Phuong-Thai II-234
 Nguyen, Quan Hoang I-265
 Novák, Attila I-360, I-399
 Novák, Borbála I-360, I-399

O

Ochs, Magalie II-329
 Ogier, Jean-Marc I-193
 Othmane Zribi, Chiraz Ben II-209

P

Pakray, Partha I-299, II-364
 Parmentier, Yannick II-209
 Parupalli, Sreekavitha I-250, II-32
 Patton, Robert II-431
 Pereira, Bolivar II-197
 Pham, Van-Lam II-234
 Phan, Viet-Anh II-341
 Piccardi, Massimo II-314
 Pisarevskaya, Dina I-74
 Pistol, Ionuț I-50
 Pla, Ferran I-131
 Poria, Soujanya I-87
 Poulain D'Andecy, Vincent I-193

R

Ragusa, Edoardo II-169
 Ruan, Chong II-59

S

Sabty, Caroline I-41
 Saikh, Tanik II-221
 Santos, Henrique D. P. II-197
 Seifollahi, Sattar II-314
 Setya, Ken Nabila I-416
 Shrivastava, Manish I-371, II-104, II-115,
 II-129, II-301
 Shushkevich, Elena II-412

Singh, Anil Kumar I-382
Singh, Kunal II-142
Singh, Navjyoti I-236, I-250, II-32
Singh, Rajat I-371, II-104, II-115, II-129,
II-301
Skitalinskaya, Gabriella II-275, II-412
Solnyshkina, Marina I-168
Solovyev, Valery I-168
Souza, Marlo II-197
Specht, Günther I-320, II-287
Stahl, Christopher II-431
Sudhakar, Akhilesh I-382
Suerdem, Ahmet II-46
Sugaya, Fumiaki I-289, II-181
Şulea, Octavia-Maria I-408

T

Takai, Kohichi I-30, I-289
Takai, Koichi II-181
Timoshenko, Svetlana I-168
Togneri, Roberto II-3
Tran, Duc-Vu II-341
Tran, Viet Hong I-265
Tripathi, Samiksha I-331
Tschuggnall, Michael I-320, II-287

U

Uban, Ana Sabina I-110
Uthayasanker, R. T. I-279

V

Van Nguyen, Vinh I-265
Vanetik, Natalia II-261
Vieira, Renata II-197
Vu, Van-Chung II-354
Vu, Xuan-Son I-153

W

Wadbude, Rahul II-71
Wang, Lin I-143
Way, Andy II-161
Wells, Jack II-431
White, Lyndon II-3
Wohlgenannt, Gerhard I-207

X

Xu, Feiyu II-381

Y

Yashothara, S. I-279
Yasuda, Keiji I-30, I-289, II-181
Yıldırım, Savaş II-46
Yoneyama, Akio I-30, II-181
Yvon, François I-349

Z

Zare Borzeshi, Ehsan II-314
Zarrabi, Vahid I-99
Zhang, Yanjun II-17
Zimmermann, Albrecht I-180
Zimmermann, Roger II-142
Zunino, Rodolfo II-169