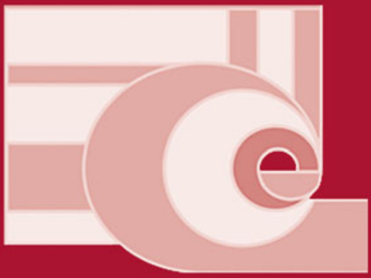
Alexander Gelbukh (Ed.)

LNCS 13396

Computational Linguistics and Intelligent Text Processing

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



Springer

Alexander Gelbukh
Editor

Computational Linguistics and Intelligent Text Processing

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

Springer

Editor
Alexander Gelbukh
Instituto Politécnico Nacional
Mexico City, Mexico

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Computer Science
ISBN 978-3-031-23792-8 ISBN 978-3-031-23793-5 (eBook)
https://doi.org/10.1007/978-3-031-23793-5

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 Politécnica de Valencia, Spain), Lucia Specia (University of Sheffield, UK), and Carlo Strapparava (Foundazione 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
Tarık 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
Yorick Wilks	University of Sheffield, UK
Vasile Rus	University of Memphis, USA

Christian Boitet	Université Grenoble Alpes, France
Khaled Shaalan	The British University in Dubai, UAE
Philipp Koehn	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 I

Contents – Part II

General

Combining Graph-Based Dependency Features with Convolutional Neural Network for Answer Triggering

Deepak Gupta[1(✉)], Sarah Kohail[2], and Pushpak Bhattacharyya[1]

[1] Indian Institute of Technology Patna, Patna, India
{deepak.pcs16,pb}@iitp.ac.in
[2] Universität Hamburg, Hamburg, Germany
kohail@informatik.uni-hamburg.de

Abstract. Answer triggering is the task of selecting the best-suited answer for a given question from a set of candidate answers if it exists. This paper presents a hybrid deep learning model for answer triggering, which combines several dependency graph-based alignment features, namely graph edit distance, graph-based similarity, and dependency graph coverage, with dense vector embeddings from a Convolutional Neural Network (CNN). Our experiments on the WikiQA dataset show that such a combination can more accurately trigger a candidate answer compared to the previous state-of-the-art models. Comparative study on WikiQA data set shows 5.86% absolute F-score improvement at the question level.

Keywords: Answer triggering · Dependency parsing · Convolutional neural network · Question answering

1 Introduction

Answer triggering is a relatively new problem for open-domain question answering (QA). In addition to extracting correct answers from a set of pre-selected candidate pools (i.e., answer selection), answer triggering detects whether a correct answer exists in the first place [12,25,33].

To evaluate the performance of answer sentence selection, WikiQA dataset has been widely used. It consists of questions collected from the user logs of the Bing search engine. The dataset is constructed using a more natural process, and it also includes questions for which there exists no correct answer. The lexical similarity between a question and answer pair in the WikiQA dataset is also lower as compared to other answer sentences selection datasets like the dataset provided by TREC QA[1] and QASent. In some cases, there is no lexical overlap at all, as shown in the following example. Given a question **Q** and correct answer **A**. **Q** and **A** do not follow any lexical similarity.

Q: *what can sql 2005 do?*
A: *As a database, it is a software product whose primary function is to store and retrieve data as requested by other software applications.*

[1] https://trec.nist.gov.

This makes the answer triggering task more challenging than the usual answer sentence selection.
Applying deep-neural-network-based models has shown a significant progress in the absence of lexical overlap between question and answer [6,30,34]. However, such models still ignore the importance of grammatical and structural relations in the context of this task.

In this paper, we propose an effective model for answer triggering, which **(i)** detects whether there exists at least one correct answer in the set of candidate answers for a given target question, in the absence of explicit lexical overlap, and **(ii)** finds the most appropriate answer from a set of candidate answers if there exists any such. Our contribution handles both – fuzzy lexical matching via Convolutional Neural Network (CNN) and grammatical structure matching via encoding dependency graphs overlap. The CNN can capture the semantic similarity between question and answer, whereas dependency graph-based features capture structural overlap resp. divergence where lexical similarity is high. We perform experiments on WikiQA dataset [33] and show that introducing graph-based features into CNN performs superior as compared to CNN alone, significantly outperform previous state-of-the-art methods.

2 Related Work

The complexity of question answering research has been increased in recent years. Due to the wider success of deep learning based model in other NLP area such as named entity recognition (NER) [8,18,24,26], sentiment analysis [10,28,32] and parsing [27,29], several deep learning based model [5,21,31] has been used to solve the Q/A problem. learn to match questions with answers by two model bag-of-words and biagram using convolutional neural network with a single convolution layer, average pooling and logistic regression. [13] present *qanta*, a dependency-tree recursive neural network for factoid question answering which effectively learns word and phrase-level representations. Convolutional neural network based deep learning model is very popular in Q/A, its success has been reported by [2,9,35–38]. Recently deep neural variational inference [19] present for answer sentence selection. [33] and [14] proposed a CNN-based model for answer triggering. However, our CNN-inspired model differs from them to calculate the semantic similarity between question and answer by means of dependency graph similarity, matching, and coverage.

3 Hybrid Model for Answer Triggering

Our method uses Convolutional Neural Network (CNN) to extract deep generic features from question-answer pairs. Our CNN maps each input sentence into a vector space, preserving syntactic and semantic aspects, which enables it to generate an effective and diverse set of features [2,3,15].

3.1 Convolutional Neural Network (CNN) Model

A simple CNN model takes a sentence as an input and performs convolution followed by pooling and classifying the sentence into one of the predefined classes by a soft-max classifier. A Joint-CNN is an advancement where question and candidate answers are input to the model. The convolution and pooling operations for questions and answers are performed separately. Thereafter, the outputs of fully connected layers (for question and answer) are concatenated to form a single input to the soft-max classification layer. The Joint-CNN model provides a probabilistic score as an output. It is inspired by the Yoon Kim [15] CNN architecture for text classification. We describe model components in the following.

Question/Answer Representation Matrix. Given a question Q and candidate answer A, having n_Q and n_A number of token respectively, each token $t_i \in Q$ and $t_j \in A$ is represented by k dimension distributed representation $x \in \mathbb{R}^k$ and $y \in \mathbb{R}^k$ respectively. The question and answer representation matrices can be generated by concatenating column level, $1 \times k$ dimensional word vector to form a $n_Q \times k$ and $n_A \times k$ dimensional matrix. We set a maximum number of tokens[2] to create a question and answer matrix.

Convolution and Pooling. To extract common patterns throughout the training set, we use the convolution operation using different feature detector (filter) lengths [15] on the question/answer representation matrix. We also apply the max pooling operation over the feature map similar to [3] on the convolution output of both question and answers. In our experiment, we used a two-layer of convolution followed by a pooling layer.

Fully Connected Layer. Finally, outputs of pooling layers p_Q and p_R are concatenated, and this resulting pooling layer $p = p_Q \otimes p_A$ is subjected to a fully connected softmax layer. It computes the probability score over two label-pair *viz* trigger, non-trigger as:

$$\begin{aligned} P(c = l|Q, A, p, a) &= softmax_l(p^T \mathbf{w} + a) \\ &= \frac{e^{p^T w_l + a_l}}{\sum_{k=1}^{K} e^{p^T w_k + a_k}} \end{aligned} \tag{1}$$

where a_k and w_k represent the bias and weight vector, respectively, of the k^{th} label.

3.2 Dependency Graph-Based Features

Both question and answer are converted into a graph using the dependency relations obtained from the Stanford dependency parser[3], following [16]. Dependency graphs of question and answer share common subgraphs of dependency links between words (c.f. Fig. 2 for an example). Based on these graphs, we extract three sets of features: graph edit distance, similarity features and coverage features.

Graph Edit Distance. Graph edit distance defines the cost of the least expensive sequence of edit operations that are needed to transform one graph, in our case dependency parsing tree, into another. It calculates the minimum cost required to transform

[2] We set maximum 20 and 40 tokens for the question and answer sentence, respectively.

[3] http://nlp.stanford.edu:8080/parser/.

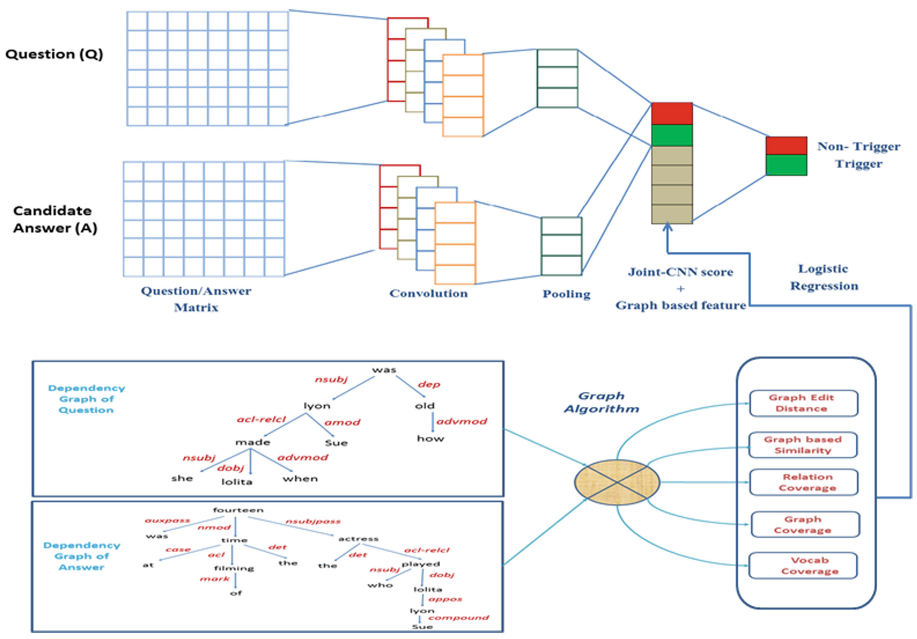


Fig. 1. Proposed model architecture for answer triggering. The architecture combines mainly two component Joint-CNN and dependency graph based alignment features. Both component works independently by taking a question and answer as input. Logistic regression is used to predicted the final label, either *'Trigger'* or *'Non-Trigger'*.

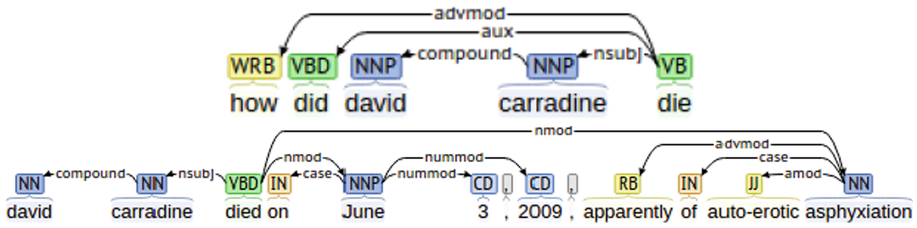

how did david carradine die
david carradine died on June 3 , 2009 , apparently of auto-erotic asphyxiation

Fig. 2. Dependency graph of a question **Q:** *"how did david carradine die"* and their correct answer **A:** *"david carradine died on June 3, 2009, apparently of auto-erotic asphyxiation"*. The word *'david'* and *'carradine'* have the same dependency relation *'compound'*. The dependency link between word *'die'* and *'how'* in question and *'die'* and *'asphyxiation'* in answer provide the similarity and coverage between question and answer.

the question graph to an answer graph. Table 1 shows the effectiveness of this feature to determine the correct answer from the pool of candidate answers. We calculate the node and edge difference between the dependency graph of question and answer. In node difference, if the two nodes from question and answer have the same word (lemma), then node difference will be '0'. Given the different words, the node difference is calculated by parts of speech (POS) substitute weight. The difference between the two POS tags is measured by substitute weight. For instance, replacing a noun with a verb should be more costly than replacing a verb with a verb. Similar to node difference, the edge (dependency relation) difference between question and answer is calculated. A two-dimensional cost matrix can be created by considering the graph edit distance between question and answer, which represents the cost of each possible node edit operation. Finally, the optimal cost is obtained by assignment algorithm [20].

Table 1. Graph-edit distance between a question and candidate answer pair. The answer which is in **bold** is the correct answer for question.

Question	Candidate Answer	Graph-Edit Distance
how old was sue lyon when she made lolita?	Lolita is a 1962 comedy-drama film by Stanley Kubrick based on the classic novel of the same title by Vladimir Nabokov.	0.98
	The actress who played Lolita, Sue Lyon, was fourteen at the time of filming	**0.59**
	Kubrick later commented that, had he realized how severe the censorship limitations were going to be, he probably never would have made the film	0.71

Dependency Graph Based Similarity. For each sentence[4] S, we define the dependency graph $G_S = \{V_S, E_S\}$, where $V_S = \{t_1, t_2, \ldots, t_{n_S}\}$ represent the tokens in a sentence, and E_S is a set of edges. Each edge e_{ij} represents a directed dependency relation between t_i and t_j. We calculate TF-IDF [23] three levels and weight our dependency graph using the following conditions:
Word TF-IDF: Consider only those words that satisfy a criteria α_1. TF-IDF $(S,\ t_i) > \alpha_1$
Pair TF-IDF: Word pairs are filtered based on the criteria α_2. TF-IDF $(S, t_i, t_j) > \alpha_2$
Triplet TF-IDF: Consider only those triplet (word, pair and relation), which satisfies a condition α_3. TF-IDF $(S, t_i, t_j, e_{ij}) > \alpha_3$
Similarities are then measured on three levels by representing each sentence as a vector of words, pairs and triples, where each entry in one vector is weighted with the TF-DF measure. The IDF is computed using the NYT part of the Gigaword corpus [7].

Dependency Graph-Based Coverage. To overcome the bias of higher similarity values between longer sentences [1], we use the coverage score between the dependency graphs of question and answer. Let $G_Q = \{V_Q, E_Q\}$ and $G_A = \{V_A, E_A\}$ be the dependency graphs of a pair of question and candidate answer. Intuitively, coverage models the fraction of the question that the answer addresses.

[4] Sentence is either question or answer.

Algorithm 1: Pseudo-code of Dependency sub-graph approximate alignment

```
Input: Dependency graph $G_Q$ and $G_A$ and threshold $m$
Output: Dependency sub-graph $G_{Sub}$
begin
    $V_{common} \leftarrow \{V_Q \cap V_A\}$ ;
    for $i = 1$ to $|V_{common}| - 1$ do
        for $j = i + 1$ to $|V_{common}|$ do
            $NodePath = FindPath(G_A, t_i, t_j)$ ;
            if $NodePath \neq \phi \wedge size(NodePath, t_i, t_j) \leq m$ then
                $G_{Sub} \leftarrow G_{Sub} \cup NodePath$ ;
            end
        end
    end
end
return $G_{Sub}$
```

Relation Coverage. We compute the number of one-to-one edge correspondence between the dependency graph of question Q and answer A, divided by the total number of edges in the dependency graph of question Q.

Graph Coverage. The idea is to find a subgraph G_{Sub} in the candidate answer dependency graph G_A that is similar to a given query text dependency graph G_Q. We use the dependency sub-graph approximate alignment algorithm by [17]. The pseudo-code of the algorithm is listed in Algorithm 1. The algorithm obtains the common set of nodes between G_Q and G_A and finds the shortest path between every pair of nodes belonging to the intersection set in the candidate answer dependency graph using Dijkstra's algorithm [4]. Each edge is assigned to a weight of 1, and edges directions are ignored during the process of the algorithm. A threshold parameter m is defined, which allows for node gaps and mismatches in the case where some nodes in the answer text cannot be mapped to any nodes in the question graph. If the shortest path size (i.e., number of edges between a pair of nodes) is less than or equal to m the path will be added to the sub-graph G_{Sub}. There are two coverage features computed on the sub-graph.

– Ratio of relation overlaps in sub-graph with respect to answer sentence dependency graph.
– Ratio of relation overlap in sub-graph with respect to question sentence dependency graph.

Vocabulary Coverage. We compute the number of one-to-one node correspondence between the dependency graph of question Q and answer A, divided by the total number of nodes in the dependency graph of question Q.

Algorithm 2:

```
procedure:FindPath(Graph $G_A$, Vertex $s$, vertex $d$)
begin
    computePath(Graph $G_A$, Vertex $s$)
    FindPathTo(Graph $G_A$, Vertex $d$)
end
```

Algorithm 3: Pseudo-code of calculating shortest path from source s in candidate dependency graph G_A

```
procedure:computePath(Graph $G_A$, Vertex $s$)
begin
    for each vertex $v \in V_A$ do
        $v.minDistance = \infty$
        $v.parent = NULL$
    end
    $s.minDistance = 0$
    Initialize a priority queue vertexQueue
    vertexQueue.$add(s)$
    while vertexQueue is not empty do
        $u$=vertexQueue.$removeHead()$; for each edge e=(u,v) do
            $temp = u.minDistance + 1$ ;
            if $temp < v.minDistance$ then
                vertexQueue.$remove(v)$
                $v.minDistance = temp$
                $v.parent = u$
                vertexQueue.$add(v)$
            end
        end
    end
end
```

4 Datasets, Experiments and Analysis

4.1 Datasets and Experimental Setup

We use the WikiQA data set for our experiments. Statistics of question and answer pairs of WikiQA data set are given in Table 2. For training the Joint-CNN model as discussed in Sect. 3, we employ stochastic gradient descent (SGD) over mini-batch, and back-propagation [11] to compute the gradients. For word embeddings, we use the pre-trained Google word embedding model[5]. The Ada-delta [39] update rule is used to tune the learning rate. The optimal hyper-parameters[6] are determined on the development data and listed in Table 3. In our final model, we embed probabilistic scores

[5] https://code.google.com/archive/p/word2vec/.

[6] Feature maps size = 100, drop-out rate = 0.5, maximum epochs = 50, learning rate = 0.2, filter window size = 3, 4, $\alpha_1 = 7$, $\alpha_2 = 5$, $\alpha_3 = 2$, $m = 2$.

Algorithm 4: Pseudo-code of finding the shortest path from source to destination

```
procedure:FindPathTo(Graph $G_A$,Vertex $d$)
begin
    Initialize a path list from source to destination $d$
    path = {}
    $vertex = d$
    while vertex is not empty do
        path.$add(vertex)$
        $vertex = vertex.parent$
    end
    path.$reverse()$
end
return path
```

obtained from CNN along with graph based linguistic features to train a logistic regression classifier. The proposed model architecture depicted in Fig. 1.

Table 2. Statistics of WikiQA data

	Train	Dev	Test
No. of Questions	2,118	296	633
No. of Answers	1,040	140	293
No. of Question w/o Answer	1,245	170	390

4.2 Baselines

We evaluate the model using information retrieval (IR) and semantic composition-based similarity. The following baselines are then used to evaluate the answer sentence selection and answer triggering task.

– **Baseline-1**: The first baseline is constructed based on the similarity measure using Okapi BM25 algorithm [22]. Each candidate answer is treated as a single document. We calculate the BM25 score between question Q and a candidate answer A. The score of a candidate answer A for a given question Q consisting of the words $q_1, ..., q_n$ is computed as:

$$\text{Score}(Q, A) = \sum_{i=1}^{n} \text{IDF}(q_i) \cdot \frac{f(q_i, A) \cdot (k_1 + 1)}{f(q_i, A) + k_1 \cdot (1 - b + b \cdot \frac{|A|}{\text{avgdl}})} \tag{2}$$

where $f(q_i, A)$ is q_i's term frequency in the candidate answer A, $|A|$ is the length of the candidate answer (in words), and $avgdl$ is the average candidate answer length in the answer pool. k_1 and b are the free parameters.

$$\text{IDF}(q_i) = log \frac{N - n(q_i) + 0.5}{n(q_i) + 0.5} \tag{3}$$

where N is the total number of candidate answers in the answer pool, $n(q_i)$ is the number of candidate answers containing q_i.
An optimal threshold value is estimated from the development data. The candidate answer which is having the score above a certain threshold value is set to '1'(triggered) and the rest are set to '0'(non-triggered).

– **Baseline-2**: Our second baseline is based on the n-gram coverage between question and answer. We compute the n-gram coverage upto 3-gram. Finally, the n-gram score between a question and an answer is calculated based on the following formula.

$$NGCoverage(Q, A, n) = \frac{\sum_{ng_n \in A} Count_{common}(ng_n)}{\sum_{ng_n \in Q} Count_{ques}(ng_n)} \tag{4}$$

$$NGScore(Q, A) = \sum_{i=1}^{n} \frac{NGCoverage(Q, A, i)}{\sum_{i=1}^{n} i} \tag{5}$$

We set a threshold value similar to the first baseline. The candidate answer which is having the score above a threshold value is set to '1'(triggered) and the rest are set to '0'(non-triggered).

– **Baseline-3**: We perform experiments using two sets of pre-trained deep learning (DL) based word embeddings, Google's word2vec embeddings of dimension 300[7] and GloVe word embeddings, of dimension 100[8]. The question/answer vector is computed as follows,

$$\texttt{VEC}(S) = \frac{\sum_{t_i \in S} \texttt{VEC}(t_i)}{\textit{number of look-ups}} \tag{6}$$

where S is question/answer in interest, *number of look-ups* represents the number of words in the question for which word embeddings are available. The cosine similarity between question vector and candidate answer vector are computed. An optimal threshold value of cosine similarity is estimated from the development data. The candidate answer having cosine similarity above the threshold score (0.70) is set to '1' (triggered), and all others are set to '0'(non-triggered).

4.3 Result and Analysis

Mean Average Precision (MAP) and Mean Reciprocal Rank (MRR) are used to evaluate the performance for answer sentence selection. But both are not suitable for evaluating the task of answer triggering because these evaluate the relative ranks of correct answers in the candidates of a question. We use the standard precision, recall, and F-score to evaluate the answer triggering problem following [33]. While evaluating, we consider all the candidate answers that yield the highest model score. If the score is above a predefined threshold, then the candidate answer is labeled as a correct answer to the question. The optimal threshold value (0.14) is determined based on the development

7 https://code.google.com/archive/p/word2vec/.
8 https://nlp.stanford.edu/projects/glove/.

data. We define three baselines, BM-25, N-Gram coverage, and semantic similarity-based model, to compare against our proposed model.

We conduct experiments with different feature map sizes for Joint-CNN. We also analyze the impact of different graph-based features. Detailed comparisons and the impact of these features are reported in Table 4. We observe the impact of the dependency graph feature in determining the suitable answers from a collection of answer pools. The feature ablation study reveals the importance of each dependency graph feature on validation and test set. However, a similar impact of each feature could not observed on the test data set. The final model comprised of the best Joint-CNN model (100-FMap) with graph edit distance, graph similarity, and graph coverage. We observe that Joint-CNN with graph-based feature achieves an improvement of 6.17, 4.21, and 3.41 points over the Joint-CNN model (without graph feature) with respect to F-score, MAP, and MRR. The best combination is obtained with a CNN that maximizes recall; the graph-based features substantially improve precision for the maximal F-score, MRR and MAP. [33] and [14] used the same WikiQA dataset to evaluate their system performance on answer triggering task. The [33] model is based on the augmentation of question class and sentence length feature to CNN. A subtree matching algorithm along with CNN architecture is used in [14] to evaluate answer triggering. Our proposed model is different from these state-of-the-art models in terms of investigation of richer linguistic features (coverage, similarity) and graph-based similarity in conjunction with the CNN model. The values obtained through the t-test show that performance improvements in our proposed model over these two state-of-the-art systems are statistically significant ($p < 0.05$). In Table 5 we provide analysis with proper examples for our two proposed models, *viz.* Joint-CNN and Hybrid.

Table 3. Neural network hyper-parameters

Parameter	Description	Value
d^x	Word embedding dimension	300
n	Maximum length of comments	50
m	Filter window sizes	3,4
$\boldsymbol{c}$	Maximum feature maps	100
r	Dropout rate	0.5
e	Maximum epochs	50
mb	Mini-batch size	50
λ	Learning rate	0.2
α_1	Threshold to filter word TF-IDF	7
α_2	Threshold to filter Pair TF-IDF	5
α_3	Threshold to filter Triplet TF-IDF	2
m	Threshold for sub-graph alignment	3

– **Q:** *What is adoration catholic church?*
  $\mathbf{A_1}$: *Adoration is a sign of devotion to and worship of Jesus Christ, who is believed by Catholics to be present Body, Blood, Soul, and Divinity.*
  $\mathbf{A_2}$: *Eucharistic adoration is a practice in the Roman Catholic Church and in a few*

Table 4. Evaluation results of answer triggering on the development and test set of WikiQA dataset: Question-level precision, recall and F-scores. MAP and MRR are used to evaluate the performance of answer selection. **FMap** denotes the size of feature map. Precision, Recall and F-score are given in percentages (%).

	Test set					Development set				
Model	MAP	MRR	Precision	Recall	F-score	MAP	MRR	Precision	Recall	F-score
Baselines										
BM-25	0.4712	0.4889	21.04	24.43	22.60	0.4569	0.4701	20.32	26.19	22.88
N-Gram Coverage	0.5102	0.5349	24.47	28.59	25.24	0.5289	0.4909	25.73	29.11	27.31
Semantic Vec (W2V)	0.4323	0.4411	13.37	34.16	19.21	0.4429	0.4395	14.55	35.87	20.70
Semantic Vec (Glove)	0.4928	0.5277	16.12	40.74	23.10	0.4987	0.4901	17.56	39.19	24.25
Our Models										
Joint-CNN (50-FMap)	0.6218	0.6322	28.90	31.28	30.04	0.6123	0.6520	33.01	26.98	29.69
Joint-CNN (150-FMap)	0.6369	0.6535	25.07	39.51	30.67	0.6152	0.6245	25.29	34.13	29.05
Joint-CNN (100-FMap)	0.6372	0.6567	23.21	**48.15**	31.33	0.6220	0.6250	25.33	46.03	32.68
+ Graph Edit Distance	0.6540	0.6708	33.18	30.04	31.53	0.6487	0.6522	32.53	36.28	34.30
+ Graph Similarity	0.6648	0.6828	25.00	43.21	31.67	0.6810	0.6744	34.85	36.51	35.66
+ Graph Coverage	**0.6793**	**0.6908**	**35.69**	39.51	**37.50**	0.6917	0.6940	39.83	37.30	38.52
State-of-the art (Answer Triggering)										
CNN-cnt+All [33]	–	–	28.34	35.80	31.64	–	–	–	–	–
CNN_3: max + emb+ [14]	–	–	29.43	48.56	36.65	–	–	–	–	–

Table 5. Comparative analysis of the result from Joint-CNN and Hybrid model on pair of questions and answers. The correct answer are in **bold**. Here **TG**: trigger and **NTG**: non-trigger model predictions, marked as correct '✓' or incorrect '✗'.

Question	Answer	Joint-CNN	Hybrid
What is adoration catholic church ?	Adoration is a sign of devotion to and worship of Jesus Christ, who is believed by Catholics to be present Body, Blood, Soul, and Divinity.	TG (✗)	NTG (✓)
	Eucharistic adoration is a practice in the Roman Catholic Church, and in a few Anglican and Lutheran churches, in which the Blessed Sacrament is exposed and adored by the faithful.	NTG (✗)	TG (✓)
where is La Palma africa?	La Palma has an area of 706 km making it the fifth largest of the seven main Canary Islands.	TG (✗)	NTG (✓)
	La Palma is the most north-westerly of the Canary Islands.	NTG (✗)	TG (✓)
What are land parcels	**land lot, a piece of land;**	NTG (✗)	TG (✓)
	fluid parcel, a concept in fluid dynamics	TG	NTG
What bacteria grow on macconkey agar	**MacConkey agar is a culture medium designed to grow Gram-negative bacteria and differentiate them for lactose fermentation.**	TG (✓)	NTG (✗)
How much is centavos in Mexico	**The peso is subdivided into 100 centavos.**	NTG (✗)	NTG (✗)

Anglican and Lutheran churches, in which the Blessed Sacrament is exposed and adored by the faithful.
Here, the Joint-CNN model selects the answer as $\mathbf{A_1}$, but the hybrid model selects $\mathbf{A_2}$, which is correct. The reason could be that vocab and graph coverage[9] between $\mathbf{Q}$ and $\mathbf{A_2}$ are higher than $\mathbf{Q}$ and $\mathbf{A_1}$.

– **Q:** *where is La Palma africa?*
$\mathbf{A_1}$: *La Palma has an area of 706 km2, making it the fifth largest of the seven main Canary Islands.*
$\mathbf{A_2}$: *La Palma is the most north-westerly of the Canary Islands.*
Both $\mathbf{A_1}$ and $\mathbf{A_2}$ are the correct answers for the question $\mathbf{Q}$, but $\mathbf{A_2}$ has more precise information compared to $\mathbf{A_1}$. Joint-CNN model selects $\mathbf{A_1}$ as correct answer, whereas the hybrid model selects $\mathbf{A_2}$, because of the higher graph coverage score.

5 Conclusion

In this paper, we have proposed a hybrid model for answer triggering using deep learning and graph-based features. Modeling QA pair is a more complex task than classifying a single sentence. It is observed that CNN does not capture the important features spanning between the text, such as quantification of similarity/dissimilarity, geometric similarity. To overcome this limitation, we investigate the incorporation of richer linguistic features (dependency graph) in CNN. Experiments on the WikiQA benchmark dataset show that integrating graph-based alignment features with CNN improves the performance significantly. Future work includes building an end-to-end neural network that can embed graph-based features with a sentence encoder (CNN, LSTM, etc.).

References

1. Albalate, A., Minker, W.: Semi-Supervised and Unervised Machine Learning: Novel Strategies. John Wiley & Sons (2013)
2. Blunsom, P., Grefenstette, E., Kalchbrenner, N.: A convolutional neural network for modelling sentences. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 655–665. Association for Computational Linguistics, Baltimore, Maryland, June 2014. http://www.aclweb.org/anthology/P14-1062
3. Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P.: Natural language processing (almost) from scratch. J. Mach. Learn. Res. **12**, 2493–2537 (2011)
4. Dijkstra, E.W.: A note on two problems in connexion with graphs. Numer. Math. **1**(1), 269–271 (1959)
5. Dong, L., Wei, F., Zhou, M., Xu, K.: Question answering over freebase with multi-column convolutional neural networks. In: ACL (1), pp. 260–269 (2015)
6. Feng, M., Xiang, B., Glass, M.R., Wang, L., Zhou, B.: Applying deep learning to answer selection: a study and an open task. In: 2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU), pp. 813–820. IEEE (2015)
7. Graff, D., Cieri, C.: English gigaword, ldc catalog no. LDC2003T05. Linguistic Data Consortium, University of Pennsylvania (2003)

[9] *Church* and *adoration* are common node between question and answer which increases the graph coverage.

8. Gupta, D., Ekbal, A., Bhattacharyya, P.: A Deep Neural Network based Approach for Entity Extraction in Code-Mixed Indian Social Media Text. In: chair), N.C.C., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Hasida, K., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., Piperidis, S., Tokunaga, T. (eds.) Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018). European Language Resources Association (ELRA), Miyazaki, Japan, 7–12 May 2018 (2018)
9. Gupta, D., Kumari, S., Ekbal, A., Bhattacharyya, P.: MMQA: A Multi-domain Multi-lingual Question-Answering Framework for English and Hindi. In: chair), N.C.C., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Hasida, K., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., Piperidis, S., Tokunaga, T. (eds.) Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018). European Language Resources Association (ELRA), Miyazaki, Japan, May 7–12, 2018 (2018)
10. Gupta, D., Lamba, A., Ekbal, A., Bhattacharyya, P.: Opinion mining in a code-mixed environment: a case study with government portals. In: Proceedings of the 13th International Conference on Natural Language Processing, pp. 249–258. NLP Association of India, Varanasi, India, December 2016. http://www.aclweb.org/anthology/W/W16/W16-6331
11. Hecht-Nielsen, R.: Theory of the backpropagation neural network. In: International Joint Conference on Neural Networks, 1989. IJCNN, pp. 593–605. IEEE (1989)
12. Heilman, M., Smith, N.A.: Tree edit models for recognizing textual entailments, paraphrases, and answers to questions. In: Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, pp. 1011–1019. Association for Computational Linguistics (2010)
13. Iyyer, M., Boyd-Graber, J., Claudino, L., Socher, R., Daumé III, H.: A neural network for factoid question answering over paragraphs. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 633–644. Association for Computational Linguistics, Doha, Qatar (October 2014), http://www.aclweb.org/anthology/D14-1070
14. Jurczyk, T., Zhai, M., Choi, J.D.: Selqa: A new benchmark for selection-based question answering. In: 2016 IEEE 28th International Conference on Tools with Artificial Intelligence (ICTAI), pp. 820–827 (Nov 2016)
15. 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. Association for Computational Linguistics, Doha, Qatar, October 2014. http://www.aclweb.org/anthology/D14-1181
16. Kohail, S.: Unsupervised topic-specific domain dependency graphs for aspect identification in sentiment analysis. In: Proceedings of the Student Research Workshop associated with RANLP, pp. 16–23 (2015)
17. Kohail, S., Biemann, C.: Matching, re-ranking and scoring: Learning textual similarity by incorporating dependency graph alignment and coverage features. In: 18th International Conference on Computational Linguistics and Intelligent Text Processing (2017)
18. Kumar, A., Ekbal, A., Saha, S., Bhattacharyya, P., et al.: A recurrent neural network architecture for de-identifying clinical records. In: Proceedings of the 13th International Conference on Natural Language Processing, pp. 188–197 (2016)
19. Miao, Y., Yu, L., Blunsom, P.: Neural variational inference for text processing. In: International Conference on Machine Learning, pp. 1727–1736 (2016)
20. Munkres, J.: Algorithms for the assignment and transportation problems. J. Soc. Ind. Appl. Math. **5**(1), 32–38 (1957)
21. Ren, M., Kiros, R., Zemel, R.: Exploring models and data for image question answering. In: Advances in Neural Information Processing Systems, pp. 2953–2961 (2015)
22. Robertson, S.E., Walker, S., Jones, S., Hancock-Beaulieu, M.M., Gatford, M., et al.: Okapi at trec-3. NIST SPECIAL PUBLICATION SP **109**, 109 (1995)

23. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Inf. Process. Manage. **24**(5), 513–523 (1988)
24. dos Santos, C., Guimaraes, V., Niterói, R., de Janeiro, R.: Boosting named entity recognition with neural character embeddings. In: Proceedings of NEWS 2015 The Fifth Named Entities Workshop, p. 25 (2015)
25. Severyn, A., Moschitti, A.: Automatic feature engineering for answer selection and extraction. In: EMNLP, pp. 458–467 (2013)
26. Shweta, A.E., Saha, S., Bhattacharyya, P.: Deep learning architecture for patient data de-identification in clinical records. In: Proceeding of Clinical Natural Language Processing Workshop (ClinicalNLP) at the 26th International Conference on Computational Linguistics (COLING 2016), Japan (accepted) (2016)
27. Socher, R., Bauer, J., Manning, C.D., Ng, A.Y.: Parsing with compositional vector grammars. In: Proceedings of the ACL Conference. Citeseer (2013)
28. Socher, R., Perelygin, A., Wu, J.Y., Chuang, J., Manning, C.D., Ng, A.Y., Potts, C.: Recursive deep models for semantic compositionality over a sentiment treebank. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), vol. 1631, p. 1642. Citeseer (2013)
29. Turian, J., Ratinov, L., Bengio, Y.: Word representations: a simple and general method for semi-supervised learning. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pp. 384–394. Association for Computational Linguistics (2010)
30. Wan, S., Lan, Y., Guo, J., Xu, J., Pang, L., Cheng, X.: A deep architecture for semantic matching with multiple positional sentence representations. CoRR abs/1511.08277 (2015)
31. Xiong, C., Merity, S., Socher, R.: Dynamic memory networks for visual and textual question answering. In: International Conference on Machine Learning, pp. 2397–2406 (2016)
32. Yadav, S., Ekbal, A., Saha, S., Bhattacharyya, P.: Medical Sentiment Analysis using Social Media: Towards building a Patient Assisted System. In: chair), N.C.C., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Hasida, K., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., Piperidis, S., Tokunaga, T. (eds.) Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018). European Language Resources Association (ELRA), Miyazaki, Japan, May 7–12, 2018 (2018)
33. Yang, Y., Yih, W.t., Meek, C.: Wikiqa: a challenge dataset for open-domain question answering. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pp. 2013–2018. Association for Computational Linguistics, Lisbon, Portugal, September 2015
34. Yao, X., Van Durme, B., Callison-Burch, C., Clark, P.: Answer extraction as sequence tagging with tree edit distance. In: HLT-NAACL, pp. 858–867 (2013)
35. Yih, W.t., He, X., Meek, C.: Semantic parsing for single-relation question answering. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pp. 643–648. Association for Computational Linguistics, Baltimore, Maryland, June 2014. http://www.aclweb.org/anthology/P14-2105
36. Yin, W., Ebert, S., Schütze, H.: Attention-based convolutional neural network for machine comprehension. arXiv preprint arXiv:1602.04341 (2016)
37. Yin, W., Schütze, H., Xiang, B., Zhou, B.: Abcnn: attention-based convolutional neural network for modeling sentence pairs. Trans. Assoc. Comput. Linguist. **4**, 259–272 (2016). https://transacl.org/ojs/index.php/tacl/article/view/831
38. Yu, L., Hermann, K.M., Blunsom, P., Pulman, S.: Deep learning for answer sentence selection. arXiv preprint arXiv:1412.1632 (2014)
39. Zeiler, M.D.: ADADELTA: an adaptive learning rate method. CoRR abs/1212.5701 (2012). http://arxiv.org/abs/1212.5701

Prediction of Cryptocurrency Market

Rareş Chelmuş[1](✉), Daniela Gîfu[1,2], and Adrian Iftene[1]

[1] Faculty of Computer Science, Alexandru Ioan Cuza" University, Iaşi, Romania
{rares.chelmus,daniela.gifu,adiftene}@info.uaic.ro

[2] Institute for Theoretical Computer Science, Romanian Academy –Iaşi Branch, Iaşi, Romania

Abstract. In this paper, we present a set of experiments on predicting the rise or fall of a cryptocurrency using machine learning algorithms and sentiment analysis of the afferent media (online press mainly). The machine learning part is using the data of the currencies (the prices at a specific time) to predict in a mathematical sense. The sentiment analysis of the media (articles about a cryptocurrency) will influence the mathematical prediction, depending on the feeling created around the currency. The study can be useful for entrepreneurs, investors, and normal users, to give them a clue on how to invest. Furthermore, the study is intended for research regarding natural language processing and human psychology (deducting the influence of masses through media) and also in pattern recognition.

Keywords: Machine learning · Cryptocurrency market · Online press corpus · Price prediction · Sentiment analysis

1 Introduction

Human history is filled with prediction attempts and with prophets (Nelson, 2000), but the ability to anticipate the future events of life evolution, especially in financial economics (Gifu and Cristea, 2012), is a complex task that can be solved using artificial intelligence (AI) techniques. Forecasting the market, the purpose of this paper remains an important challenge with a lot of implications for safety economy. Economists developed, through many studies and observations on graphs and data, diverse techniques to get an idea of how the prices will rise or fall (martingales) (Feller, 1971). The accuracy of data retrieval depends on the understanding of the rational, emotional and physiological limiting factors towards the stock market. There are also various variables to consider as: the causal impact of media, investor's behavior, time-varying local economic conditions, history of prices, natural calamities, etc. They can systematically affect the fluctuation of the market. Some specialists in financial economics and big data consider that trying to predict and invest in a random manner can direct to same results. This assumption is known as the EMH (*Efficient Market Hypothesis*)[1] or the RWH (*Random Walk Hypothesis*) (Fama 1965; Jonathan Clarke et al. 2001; Zunino et al. 2012; Bariviera et al. 2014; Marwala 2015; Marwala and Hurwitz 2017).

[1] Starting with Eugene Fama's PhD (1965), EMH becoming one of the most known theories in financial economics that confirm the connection between prices and public information.

A. Gelbukh (Ed.): CICLing 2018, LNCS 13396, pp. 17–29, 2023.
https://doi.org/10.1007/978-3-031-23793-5_2

In this study, we have chosen the cryptocurrency market. *Why*? Because, Bitcoin (BTC), Ethereum (ETH), Ripple (XRP) and the rest of the cryptocoins and tokens represent the future of money. Moreover, in the last year, the market capitalization of cryptocurrency has grown from 16 billion of dollars (10 January 2017) to an outstanding peak of 829 billion of dollars (7 January 2018)[2]. Because of the high trading volume, a lot of data (graphs and text) is gained everyday[3] and is free for the entire public to be used by every possible mean. The best approach is proving to be a combination of mathematical models with psychological models because it has a projected growth rate on prediction.

The paper is organized as follows: Sect. 2 shortly reviews the relevant literature on various approaches about the huge role of predicting the cryptocurrency in human life; Sect. 3 presents materials (data set) and methods based on machine learning (ML) algorithms; the results are described in Sect. 4; Sect. 5 highlights a short discussion about this survey, and, finally, our conclusions are given in Sect. 6.

2 Related Work

A collection of AI algorithms has been used to predict the fluctuation of a cryptocurrency. Some of them were developed through observation, like *mean reversion*, based on the assumption that the stocks will reach the average value in time (Spierdijk and Bikker, 2012; Dai et al., 2013). However, the problem still remains: you cannot really predict when and how long the prices will stay on the average. We know that stock prices fluctuate in a random manner (Granger 1992). It is the reason why stock investors use martingales to predict the future trend.

To forecast a linear graph, we have discovered the fact that linear regression could be intuitively used, being among the first attempts in predicting the course of a practical linear model (Xin 2009). This algorithm, one of the most well-known algorithms in machine learning, was used in solving the prediction problem due to the nature of the data set used in the stock market (graph, CSV) (Nunno 2017).

Another (supervised) machine learning algorithm used in reducing the risk of investing in the stock market is Random Forests. This model constructs a number of decision trees and gives as an output the class or mean prediction of the trees. The method has been proven effective in trading, but on a long-term period (Khaidem et al. 2016) and on the stock market. It is known the fact the stock market is a more stable market than the crypto one.

For the overreaction-driven price momentum mechanism (Daniel et al. 1998; Hou et al. 2009), media is reflected on the investors' decision behavior (Barber and Odean 2008).

In general, sentiment analysis (SA) as a NLP (*Natural Language Processing*) approach is a popular predicting algorithm. Hilbert considers that the negative news increase the probability to lower the prices of stocks. On the other hand, the positive news could reflect a rise of the value of stocks (Hilbert et al. 2014).

[2] Cryptolization.com.

[3] Coinmarketcap.com.

In our work, the linear regression, random forest and random forest aided by sentiment analysis algorithms formed the basis for the prediction models of the cryptocurrency market described below.

3 Materials and Methods

In this section, we describe the data set and methods used to predict cryptocurrency market (here 3 cryptocoins, called: Ethereum, Bitcoin and Ripple), on which we will build our prediction models.

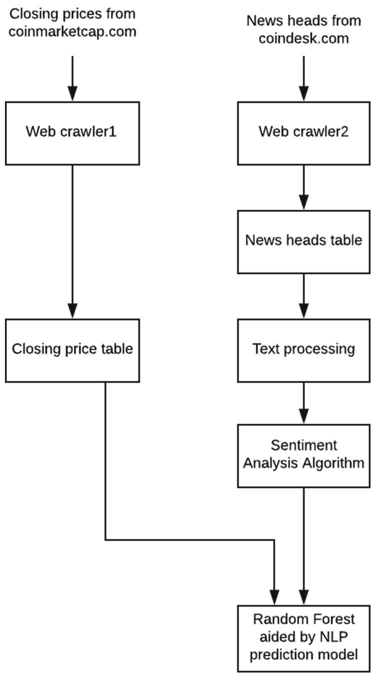

Closing prices from coinmarketcap.com
News heads from coindesk.com
Web crawler1
Web crawler2
News heads table
Closing price table
Text processing
Sentiment Analysis Algorithm
Random Forest aided by NLP prediction model

Fig. 1. Architecture of the cryptocurrency prediction

For these experiments, we preferred to choose cryptocoins that are found in the top 5 of the cryptocurrency market, because these coins are considered mature, meaning that there's a lot of trading activity every day (the data is diversified). The trading volume and, also the notoriety of the cryptocoins is direct proportional with the place held in the top. Moreover, the media closely follows how they fluctuate over time. The titles of articles have been analyzed by us using sentiment analysis classifiers from the NLTK python library. The algorithm will gather data on a period of 3 months, from 15 October 2017 to 05 January 2018. The reason for this short period of time is that the cryptomarket is still evolving and in the last 3 months in media spiked a huge flow of articles regarding cryptocurrencies.

3.1 Data Set

We test our system on two data collections: the closing price chart and the news heads table. Their differences permit us to improve the performance of our approach in types of cryptocurrency's fluctuations. This is also a great start for entrepreneurs, programmers and researchers to find better solutions on predictions, because using a site to get the news titles helps in avoiding hazard in data. Data could be scraped from other sites like Twitter, Reddit forums, YouTube and other media from where you can gather various information. However, the problem is finding the best source and cleaning the data set (the English used in forums and social media isn't all the time accurate and can have a lot of mistakes). Thus, for the moment, we will get our data from the best news site on crypto.

3.1.1 Text Data

The text data set (Table 1 contains only a sample of data) is used by the NLP component of the prediction algorithm and is formed of article titles from cryptocurrency news site[4]. Note that the articles, directed towards the coin in order to be predicted, will not appear daily. In this case, we approximate the missing scores using our formula (3), tailored for our needs. In addition, there is a possibility that many articles about the coin would appear in a single day.

Table 1. Statistical data.

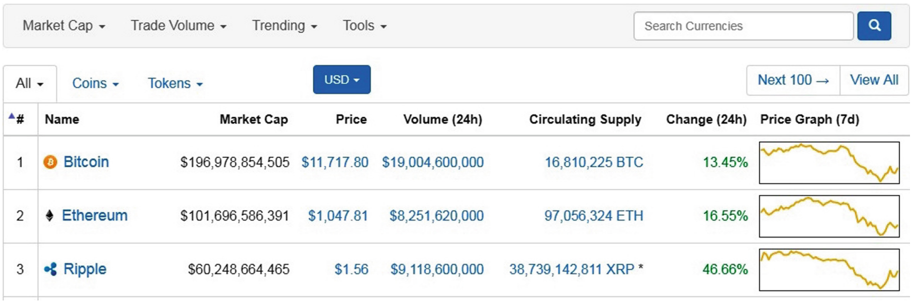
Data	Title	N words of title
15/10/2017	Hours to go: How to Watch Ethereum's Fork as It Happens	11
16/10/2017	Bulls Take Breather? Bitcoin Slows as Price Struggles to Breach $6,000	11
…	…	…
1/01/2018	What Will the Bitcoin Price Be in 2017?	8
02/01/2018	Is It Too Late to Buy Bitcoin? Video: $1 Million? Bitcoin Sign Guy on Why It's Not Too Late to Buy	21
03/01/2018	RSK Beta Brings Ethereum-Style Smart Contracts Closer to Bitcoin	10
04/01/2018	Ripple Fever? Other Crypto Assets Are Outpacing Its 2018 Gain	10
05/01/2018	Ethereum Price Highs Overshadow New Wave of Tech Issues	9
Total **Average**		**4473** **8.89**

Note that all text data set contains 420 (Bitcoin), 56 (Ethereum), and 22 (Ripple) articles

[4] www.coindesk.com.

3.1.2 Graphs Data

The price chart (see Fig. 2) is a table that consists of eight columns, but for this survey, we keep the date and the closing price of the cryptocurrency. The table has an entry for each day, on a period of approx. 3 months and the data will be fetched by a web crawler. Figure 3 presents a good image about the volatility of the cryptocurrency market.



#	Name	Market Cap	Price	Volume (24h)	Circulating Supply	Change (24h)	Price Graph (7d)
1	Bitcoin	$196,978,854,505	$11,717.80	$19,004,600,000	16,810,225 BTC	13.45%	
2	Ethereum	$101,696,586,391	$1,047.81	$8,251,620,000	97,056,324 ETH	16.55%	
3	Ripple	$60,248,664,465	$1.56	$9,118,600,000	38,739,142,811 XRP *	46.66%	

Fig. 2. The prices distributions of the 3 most mature coins per time

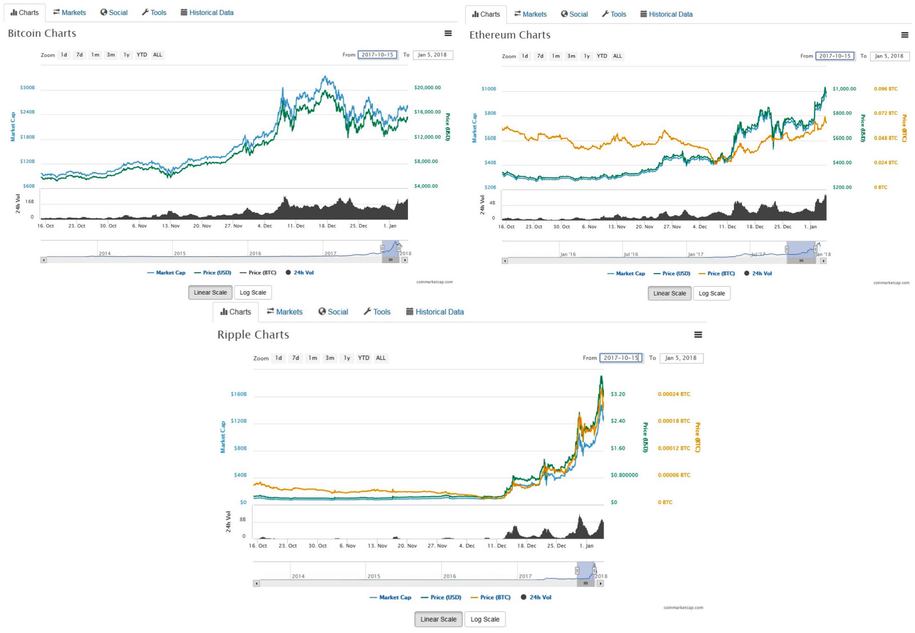


Fig. 3. Price trends of Bitcoin, Ethereum, and Ripple over the past 3 months (October 15, 2017–January 05, 2018). Data source: coinmarketcap.com

3.2 Phases Methodology

The work flow (Fig. 1) starts with the web crawlers that gather data from specific sites to fill the tables. There are two crawlers: one that scrapes a site where the closing prices of the cryptocoins are held online in a table form[5], and the other crawler gets the head of the news articles from a site specialized on cryptocurrency news[6].

The next step is to pass the news titles through the SentimentIntensityAnalyzer module from the NLTK library[7] in Python; this will be the sentiment analysis component. The title will be automatically annotated to be passed through a polarity detection algorithm regarding the feeling that is deducted. The process will give scores for each piece of text, consisting in two types of measures: positive and negative feeling score. The final score will be calculated by the formula:

$$\begin{gathered}\mathit{final\_score} = \mathtt{(sum(}\mathit{positive\_score}\mathtt{)} - \\ \mathtt{sum(}\mathit{negative\_score}\mathtt{))} \;/\; \mathit{number\_of\_articles}\end{gathered} \tag{1}$$

The sum (1) is used to solve the multiple articles problem, to answer at the question: *what sentiment is dominant?* Note that, in some days, media wrote more articles about those three cryptocoins. Averaging the sum of all positive scores minus the sum of the negative ones will result the overall sentiment on that day.

Table 2. SA scores of the media articles analyzed between 15.10.2017–05.01.2018.

Data	Scores
15/10/2017	0
16/10/2017	−0.157
16/10/2017	−0.149
16/10/2017	0
17/10/2017	−0.192
18/10/2017	0
18/10/2017	−0.208
18/10/2017	0
26/10/2017	0.426
26/10/2017	−0.216
…	…

We have three distinct situations (Table 2): 1) periods of time without data (e.g. between 2017–10-18 and 2017–10-26), 2) multiple articles in the same day (e.g. 2017–10-16), and 3) neutral articles (when the SA score is equal with 0). Because articles regarding a crypto coin do not appear every day, gaps in the table will occur.

[5] https://coinmarketcap.com.
[6] https://www.coindesk.com/.
[7] http://www.nltk.org/.

Goel from Stanford University (Mittal and Goel, 2012) filled these blank spaces with the formula (2):

$$gap = (x + y)/2 \tag{2}$$

where x and y are table boxes filled with values, and between x and y are n number of blank spaces (period of days when nothing was published towards the analyzed cryptocoins).

Moreover, machine learning could be used to predict the values for the blank spaces. However, the machine learning algorithm won't behave well because of the anomalies in the graph (unexpected rise and fall in the graph, specific for the cryptomarket). Our approach is a modification to Goel's formula:

$$gap = (x + y)/ (\mathrm{sqrt}\ (n + 1) + 1) \tag{3}$$

This alteration changes the behavior of the graphic, reflected in formula (3), having a more natural evolution from x to y. The algorithm responds better on our needs. There are three cases on how the graphic would behave (we will take $n = 5$) (Fig. 4, 5 and 6):

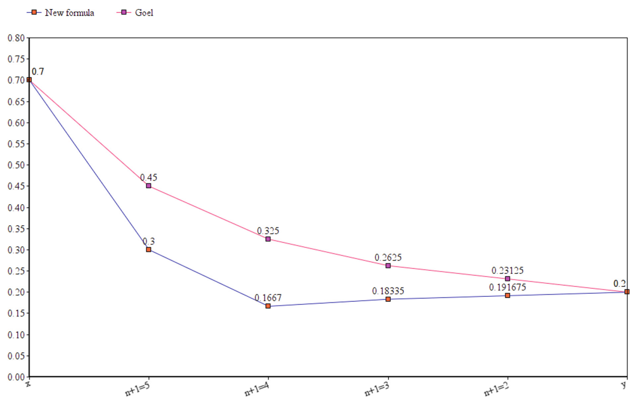


Fig. 4. Case 1: $x > y$

The formula (3), represented by the blue line, has a more natural evolution, because it falls and it also rises to reach the y value, despite Goel's equation, where it only falls.

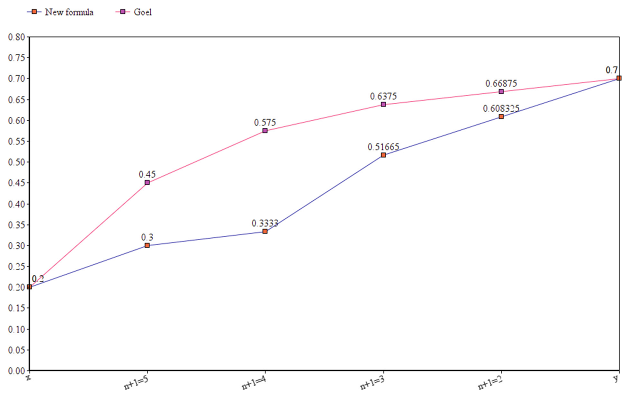


Fig. 5. Case 2: $x < y$

Note that the first rise is not as abrupt as Goel's formula ($0.45 > 2 \times x$) $\rightarrow$ responds better according with the real cryptocurrency market fluctuations.

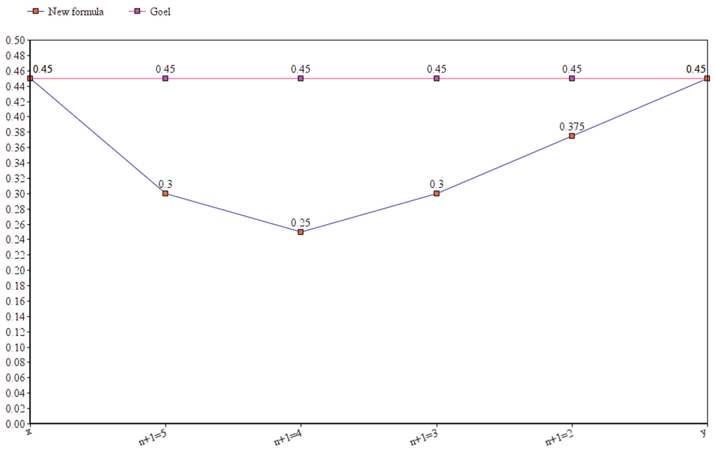


Fig. 6. Case 3: $x = y$

We can observe as well that there is a movement in the line even though x is equal with y. In Table 3, we used the formula (3), in order to complete the scores where we have no data.

Table 3. The final input for the ML algorithm

Data	Scores	Close
15/10/2017 00:00:00	0	336.6
16/10/2017 00:00:00	−0.153	333.38
17/10/2017 00:00:00	−0.192	317.08
18/10/2017 00:00:00	−0.208	314.32
19/10/2017 00:00:00	−0.0206	308.09
20/10/2017 00:00:00	0.00094	304.01
21/10/2017 00:00:00	0.0067	300.19
22/10/2017 00:00:00	0.00658	295.45
23/10/2017 00:00:00	0.00656	286.95
24/10/2017 00:00:00	0.00656	298.33
25/10/2017 00:00:00	0.02231	297.93
26/10/2017 00:00:00	0.105	0.105
…	…	…

Our method is based on three algorithms, using the scikit-learn Python library[8]: linear regression, random forests and random forests aided by sentiment analysis. In this way, we can compare the precision, the recall and the f-measure obtained with each of them, in order to analyze the potency of the NLP component.

3.2.1 Linear Regression Model

As mentioned before, Linear Regression is a primitive model used in predicting stock prices and not very efficient (because of the high fluctuation of the market). The Linear regression formula (4) usually looks like:

$$y = b_0 + b_1 \times x \tag{4}$$

where y is a dependent value (the price we want to predict), b_0 is called intercept, x the independent variable (the regressor, the price we know) and b_1 is the slope of the line.

3.2.2 Random Forests Model

Random forest works by creating a multiple of decisions trees trained by random samples of data from the data set. The result is an average or the vote of the majority of the outputs of the trees. In this case, we will only use the closing price table (only one feature). Thus the algorithm trains on 90% of the data (first 90% rows) and the last 10% will be used for test.

8 http://scikit-learn.org.

3.2.3 Sentiment Analysis with Random Forests Development

In this case, the algorithm will be trained on both tables (Table 2 and 3). According with two features (price and sentiment score) (Fig. 7) the decision trees are expected to be more ample, because of the new feature, and they will also use the same splitting scheme (first 90% for training and last 10% for testing).

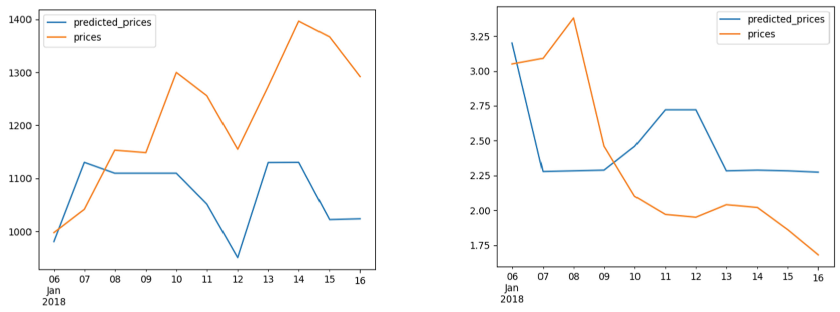


Fig. 7. The SA evolution for prices and predicted prices

The graphs above represent the output of the Random Forest, aided by Sentiment Analysis module. The orange line is the price evolution and the blue one represents the prediction of prices. The first chart is a prediction for the Ethereum data set, and the second graph was made on the Ripple data.

4 Results

From the entire set of data presented above, we looked at Precision, Recall and F-measure performance measures to evaluate our model for the prediction problem proposed in this paper (Table 4).

The results are promising. Still, we observe that through SA we gain better results, understanding that the feeling over media really influence the flow of the market. Furthermore, Random Forest works better using two features, not only one. The Linear Regression model had the worst performance on prediction, using this kind of data.

Table 4. The predictions measures of three models

Coin	Model	Precision	Recall	F-measure
	Random Forest	75.40%	65.37%	66.37%
BTC	Linear Regression	62.45%	60.08%	61.24%
	SA with Random Forests Development	75.23%	73.67%	74.44%
	Random Forest	72.11%	60.33%	61.69%
ETH	Linear Regression	62.71%	55.11%	56.85%
	SA with Random Forests Development	72.12%	71.55%	71.83%
	Random Forest	68.23%	61.42%	61.32%
XRP	Linear Regression	59.89%	54.78%	55.82%
	SA with Random Forests Development	68.45%	67.01%	67.72%

5 Discussion

This section describes the study results by running a series of three algorithms (Random Forests, Linear Regression, and SA with Random Forests Development). Our aim in these experiments was to compare three known techniques. In addition, our final goal was to define a new formula (3), by adding new missing data, in order to predict the fluctuation of a cryptocurrency. For this purpose, we compare the Precision and Recall measures.

As we observed the graphs of prediction, the nodes do not match the price very accurate. However, there's no need in matching exactly, more important is to predict the rise and fall of the price in the right time. The results showed that in the Ethereum prediction chart, a down spike occurred in the both lines at 12th January 2018. This reveals that is a great time to invest, and that the day before is perfect to sell the cryptocoins for profit. The behavior of the Ethereum prediction went well because the text data set is more than double then Ripple's media coverage. These points out the influence of the media on the investor's opinion on buying and selling on the cryptomarket. Only one problem we face: the lack of trusted media to train our algorithms.

6 Conclusions

This paper describes how machine learning algorithms and sentiment analysis of the afferent media are affecting the prediction process of cryptocurrency market. This method could become very helpful for entrepreneurs and people who would like to invest in such market and need a promising start.

Trying to predict a young market is not an easy task. The shortage of data is a big challenge. There are not many sites writing about cryptocurrency, making difficult the process of trusting. On the other hand, the multitude of unrefined sources (blogs, forums, twitter) gives us a hard time to identify feelings in comments; various heuristics need to be written in the code. Different sites may write about the same story, but can forge different feelings (humans tend to be subjective). The Efficient Market Hypothesis (EMH) seems to work here in some points of time, cryptocoins jumping on new levels of prices, becoming more and more valuable in a short time. This is the reason why the data is scraped only of the last three months and why Random Forest is used as the main algorithm (randomness seems to work).

Until the market will get in a mature phase and media will be more and more involved in the cryptomarket, the best approach in solving the gap problem is to approximate the sentiment score, using a formula. Therefore, the algorithm will better tuned, even though, big spikes (low or high) in the graph (stock or cryptocoins) still remain a hard problem to solve.

Acknowledgments. This survey was published with the support of the grant of the Romanian National Authority for Scientific Research and Innovation, CNCS/CCCDI – UEFISCDI, project number PN-III-P2–2.1-BG-2016–0390 and by a grant of the Romanian Ministry of Research and Innovation CCCDI-UEFISCDI, project number PN-III-P1–1.2-PCCDI-2017–0818 within PNCDI III.

References

Barber, B.M., Odean, T.: All that glitters: the effect of attention and news on the buying behavior of individual and institutional investors. In: Review of Financial Studies, **21**, 785–818 (2008)

Bariviera, A.F., Zunino, L., Guercio, M.B., Martinez, L.B., Rosso, O.A.: Revisiting the European sovereign bonds with a permutation-information- theory approach. Eur. Phys. J. B. **86**, 110 (2014). https://doi.org/10.1140/epjb/e2013-40660-7

Clarke, J., Jandik, T., Mandelker, G.: The efficient markets hypothesis. In: Robert C. ARFFA, ed. Expert Financial Planning: Investment Strategies from Industry Leaders. Wiley, New York Chapter 9, pp. 126–141 (2001)

Dai, Y., Zhang, Y.: Machine Learning in Stock Price Trend Forecasting. Stanford University (2013)

Daniel, K., Hirshleifer, D., Subrahmanyam, A.: Investor psychology and security market under- and overreactions. In: Journal of Finance, **53**, 1839–1885 (1998)

Fama, E.: The behavior of stock market prices. J. Bus. **38**, 34–105 (1965). https://doi.org/10.1086/294743

Feller, W.: Martingales. In: An Introduction to Probability Theory and Its Applications, Vol. 2, Wiley, New York, pp. 210-215 (1971)

Gîfu, D., Cristea, D.: Public discourse semantics. a method of anticipating economic crisis presented at the exploratory workshop on intelligent decision support systems for crisis management, 8–12 May 2012, Oradea, Romania. In: International Journal of Computers, Communications and Control, see, I. Dzitac, F.G. Filip, M.-J. Manolescu (eds.), vol. 7/5, Agora University Editing House, pp. 829–836 (2012)

Granger, C.W.J.: Forecasting stock market prices: lessons for forecasters. In: International Journal of Forecasting 8, North-Holland, pp. 3–13 (1992)

Hilbert, A., Jacobs, H., Müller, S.: Media makes momentum. Review of Financial Studies **27**(12), 3467–3501 (2014)

Hou, K., Peng, L., Xiong, W.: A Tale of Two Anomalies: The Implications of Investor Attention for Price and Earnings Momentum, SSRN 976394 (2009)

Khaidem, L., Saha, S., Dey, S. R.: Predicting the direction of stock market prices using random forest. In: Applied Mathematical Finance, pp. 1–20 (2016)

Marwala, T.: Impact of Artificial Intelligence on Economic Theory – via arXiv.org (2015)
Marwala, T., Hurwitz, E.: Artificial Intelligence and Economic Theory: Skynet in the Market. Springer, London (2017). https://doi.org/10.1007/978-3-319-66104-9
Mittal, A., Goel, A.: Stock prediction using twitter sentiment analysis. Stanford University, CS229 (2012)
Nelson, R.: Prophecy: A History of the Future - The Rex Research Civilization Kit (2000). http://www.rexresearch.com/prophist/phfcon.htm
Nunno, L.: Stock Market Price Prediction Using Linear and Polynomial Regression Models (2017)
Spierdijk, L., Bikker, J.A.: Mean Reversion in Stock Prices: Implications for Long-Term Investors (2012)
Zunino, L., Bariviera, A.F., Guercio, M.B., Martinez, L.B., Rosso, O.A.: On the efficiency of sovereign bond markets. Phys. A Stat. Mech. Appl. **391**, 4342–4349 (2012). https://doi.org/10.1016/j.physa.2012.04.009
Xin, Y.: Linear Regression Analysis: Theory and Computing (2009)

I-vectors and Deep Convolutional Neural Networks for Language Identification in Clean and Reverberant Environments

Panikos Heracleous[1], Yasser Mohammad[2,3](✉), Kohichi Takai[1], Keiji Yasuda[1], and Akio Yoneyama[1]

[1] KDDI Research, Inc., Japan, 2-1-15 Ohara, Fujimino-shi, Saitama 356-8502, Japan
{pa-heracleous,ko-takai,ke-yasuda,yoneyama}@kddi-research.jp
[2] National Institute of Advanced Industrial Science and Technology, Tokyo, Japan
y.mohammad@aist.go.jp
[3] Assiut University, Asyut, Egypt
yasserm@aun.edu.eg

Abstract. In the current study, a method for automatic language identification based on deep convolutional neural networks (DCNN) and the i-vector paradigm is proposed. Convolutional neural networks (CNN) have been successfully applied to image classification, speech emotion recognition, and facial expression recognition. In the current study, a variant of typical CNN is being applied and experimentally investigated in spoken language identification. When the proposed method was evaluated on the NIST 2015 i-vector Machine Learning Challenge task for the recognition of 50 in-set languages, a 3.9% equal error rate (EER) was achieved. The proposed method was compared to two baseline methods showing superior performance. The results obtained are very promising and show the effectiveness of using DCNN in spoken language identification. Furthermore, in the current study, a front-end feature enhancement and dereverberation approach based on a deep convolutional autoencoder is also reported.

Keywords: Spoken language identification · Deep convolutional neural networks · Dereverberation · Denoising autoencoder

1 Introduction

Automatic spoken language identification is the process of automatically recognizing language in a spoken utterance. Language identification is an important part of speech-to-speech translation systems and has a significant role in the diarization of meetings. Moreover, it can be utilized in call centers to automatically route incoming calls to appropriate native speaker operators.

Several studies have investigated spoken language identification. The approaches presented here are categorized based on the features they employ. Language identification systems are categorized into the following approaches:

acoustic-phonetic, phonotactic, prosodic, and lexical [16]. In phonotactic systems [16,28], sequences of recognized phonemes obtained from phone recognizers are modeled. In [2], a typical phonotactic language identification system is used, where a language-dependent phone recognizer is followed by parallel language models (PRLM). In [24] a universal acoustic characterization approach to spoken language recognition is proposed. The main idea is to describe any spoken language with a common set of fundamental units, such as manner and articulation, which are used to build a set of language-universal attribute models. The vector space modeling-based phonotactic language recognition approach is demonstrated in [14,16] and presented in [22]. The key idea is to vectorize a spoken utterance into a high-dimensional vector, thus leading to a vector-based classification problem.

In acoustic modeling-based systems, however, each recognized language is modeled by using different features. Although significant improvements in LID have been achieved from phonotactic approaches, most state-of-the-art systems still rely on acoustic modeling.

In [3], an early attempt at language identification based on a neural network is presented. Similarly, neural network-based language identification is addressed in [15]. In [18] the first attempt at language identification using deep learning is presented. In [13] automatic language identification based on deep neural networks (DNN) is also presented. This method demonstrates performance superior to i-vector-based [6] classification schemes when a large amount of data is used. The method is compared to linear logistic regression, linear discriminant analysis- (LDA) based, and Gaussian modeling-based classifiers. When limited training data are used, the i-vector yields the best identification rate. Another method based on DNN and using deep bottleneck features is presented in [10]. A method for identification from short utterances based on long short-term memory (LSTM) recurrent neural networks (RNN) is presented in [27]. In [5] the problem of language identification is addressed by using i-vectors with support vector machines (SVMs) [4] and LDA. SVM with local Fisher discriminant analysis is also used in [23]. Similarly to the current study, the method is evaluated on the NIST 2015 i-vector Machine Learning Challenge task. The results obtained closely resemble the results obtained in the current study when using SVM. In [20] deep neural networks-based language identification is also presented. The method is also evaluated on the NIST 2015 i-vector Machine Learning Challenge task.

In the current study, a method for automatic language identification based on the i-vector paradigm and deep convolutional neural networks (DCNN) is proposed. Convolutional neural networks [1,12] have been successfully applied to sentence classification [11], image classification [21], facial expression recognition [8], and in speech emotion recognition [17], Furthermore, in [7] bottleneck features extracted from CNN are used for robust language identification.

The motivation of using CNN, instead of the conventional fully connected feed-forward neural network, lies in the fact that CNN require less parameters. As a result is cheaper in terms of memory and compute power compared to

DNN. Furthermore, previous studies reported robustness against noise of CNN. Also, since the inputs to the network are i-vectors and not sequences, CNN offer a reliable solution to the classification problem. In the current study, DCNN, a variant of typical CNN, is used and experimentally investigated in spoken language identification.

In addition to language identification, the effectiveness of convolutional neural networks in dereverberation is also addressed. Specifically, a convolutional denoising autoencoder (DAE) [9] is used to map the reverberant speech into clean speech. DAEs are used for feature enhancement using a class of deep neural networks (DNN) and they are trained to map a corrupted speech observation to a clean one. DAEs have been successfully used in automatic speech recognition in noisy and reverberant environments, and they have been shown to improve recognition rates significantly.

2 Methods

2.1 Data

In the NIST 2015 LRE i-Vector Machine Learning Challenge task, i-vectors, constructed from conversational and narrow-band broadcast speech, are given as training, testing, and development data. The task covers 50 languages, and contains 15000 training i-vectors, 6500 test i- vectors, and 6431 development i-vectors. The training i-vectors are extracted from speech utterances with a mean duration of 35.15 s. The training data and the test data are labeled, but the development i-vectors are unlabeled. The set also includes i-vectors corresponding to out-of-set languages. In the current study, only the in-set languages are considered. In particular, 300 training i-vectors and 100 test i-vectors are used for each of the 50 in-set languages.

In the current study, *NTT-AT multilingual speech database for telephonometry 1994* was also used to investigate the effectiveness of deep convolutional denoising autoencoders in de-reverberation. The data cover 21 languages and four male and four female speakers are assigned to each language. Twenty-four short utterances (i.e., approximately 4 s) are spoken by each native speaker. Speech data are sampled at 16-bit and 16 kHz rates.

The reverberant emotional speech data are simulated based on the convolution method. Specifically, impulse responses are recorded in different environments and then convoluted with the clean data in order to create the reverberant emotional speech data. For recording, a linear microphone array with 14 transducers located at 2.83cm intervals is used [25]. The impulse response is measured using the TSP method [26]. TSP length is 65536 points and the number of synchronous additions is 16. Impulse responses in five different rooms are recorded. The $T_{[60]}$ reverberation times are 0.30, 0.47, 0.60, 0.78, and 1.3 s, respectively.

2.2 The i-Vector Paradigm

Gaussian mixture models (GMM) with universal back- ground models (UBM) are widely used for speaker recognition. In such a case, 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 model. Similar to speaker recognition, GMM supervectors can also be utilized for language identification.

The main disadvantage of GMM supervectors is the high dimensionality, which requires 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 automatic language identification, an input utterance can be modeled as:

$$\mathbf{M} = \mathbf{m} + \mathbf{T}\mathbf{w} \tag{1}$$

where $\mathbf{M}$ is the language-dependent supervector, $\mathbf{m}$ is the language-independent supervector, $\mathbf{T}$ is the total variability matrix, and $\mathbf{w}$ is the i-vector. Both the total variability matrix and language-independent supervector are estimated from the complete set of the training data.

2.3 Classification Approaches

Support Vector Machines (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, which 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 [19], 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:

$$\mathbf{w}_{i,j} = \beta + \mathbf{S}\mathbf{x}_i + \mathbf{e}_{i,j} \tag{2}$$

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 $\mathbf{\Sigma}$.

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 $\{\beta, \mathbf{S}, \mathbf{\Sigma}\}$ parameters.

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.

2.4 Evaluation Measures

In the current study, the EER (i.e., equal false alarms and false rejections), the identification rates, and the cost functions are used as evaluation measures. Considering that in the current study only the in-set languages are being recognized, the cost function defined by NIST is modified as follows:

$$C_{avg} = \frac{1}{n}\sum_{k=1}^{n} P_{error}(k) \cdot 100 \tag{3}$$

where

$$P_{error}(k) = \frac{No.\ of\ errors\ for\ class\ k}{No.\ of\ trials\ for\ class\ k} \tag{4}$$

The identification rate is defined as:

$$acc = \frac{1}{n}\sum_{k=1}^{n} \frac{No.\ of\ corrects\ for\ class\ k}{No.\ of\ trials\ for\ class\ k} \cdot 100 \tag{5}$$

where n is the number of the target languages. In addition, the detection error tradeoff (DET) curves, which show the function of miss probability and false alarms, are also given.

The effectiveness of the convolutional denoising autoencoder in feature enhancement is evaluated using the mean squared error (MSE) and cosine similarity values. The MSE is defined as follows:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y_i} - Y_i)^2 \tag{6}$$

Table 1. C_{avg} cost for different language sub-set of NIST LRE 2015.

No of languages	Classification method		
	DCNN	SVM	PLDA
10	4.2	5.4	8.2
20	9.6	10.7	16.5
30	11.7	13.9	23.6
40	13.8	15.7	27.9
50	16.0	18.6	32.9

Table 2. EER for different language sub-set of NIST LRE 2015.

No of languages	Classification method		
	DCNN	SVM	PLDA
10	1.8	2.4	3.2
20	3.7	4.1	5.5
30	3.6	4.5	6.0
40	3.4	4.7	6.3
50	3.9	5.2	6.7

The cosine similarity is given by the following formula:

$$\cos(\theta) = \frac{\sum_{i=1}^{n} Y_i \hat{Y}_i}{\sqrt{\sum_{i=1}^{n} Y_i^2}\sqrt{\sum_{i=1}^{n} \hat{Y_i}^2}} \tag{7}$$

where $\hat{\mathbf{Y}}$ is the vector of n predictions, and $\mathbf{Y}$ is the input vector, which generated the predictions.

3 Results

3.1 Language Identification Experiments

These sections present the experimental results for automatic language identification. The experimental results show the performance of the proposed method compared to SVM, and PLDA for the identification of 10, 20, 30, 40, and 50 in-set target languages using the NIST 2015 LRE i-Vector Machine Learning Challenge task. For language identification, a DCNN with four convolutional layers, one *Maxpooling* layer, and one fully connected output layer was used. The number of filters in the convolutional layers was set to 32, 128, 128, and 128,

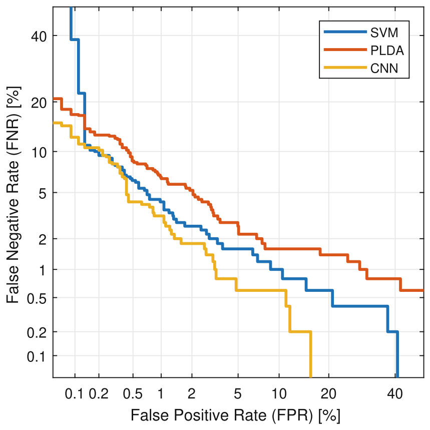


Fig. 1. DET curves for ten target languages.

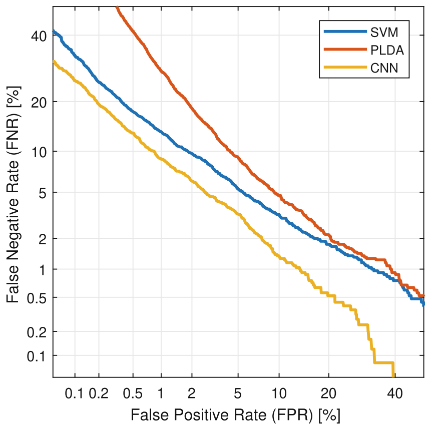


Fig. 2. DET curves for the fifty target languages.

respectively, and the epochs number was set to 150. In the convolutional layers, the $ReLu$ activation function was used, and in the fully connected output layer, the $Softmax$ activation function was chosen. Finally, the dropout probability was set to 0.15.

Table 1 shows the costs for 10, 20, 30, 40, and 50 target languages, respectively. The results show that the lowest average costs are obtained when using DCNN. It is also shown, the DCNN is followed by SVM. The results show the effectiveness of the proposed method in spoken language identification. In the case of the 50 in-set target languages, the average cost function is only 16.0%, which is a very promising result and superior to the results obtained from other similar studies. The results also show that the cost when using PLDA rapidly increases as the number of target languages increases, and that PLDA is less effective for the current task.

Table 3. Equal error rates using training data of different sizes.

No. of training	Classification method		
i-vectors	DCNN	SVM	PLDA
2500	6.2	8.3	7.2
5000	4.8	6.7	6.9
7500	4.4	6.4	7.2
10000	4.3	6.1	7.6
12500	4.0	5.8	7.3
15000	3.9	5.2	6.7

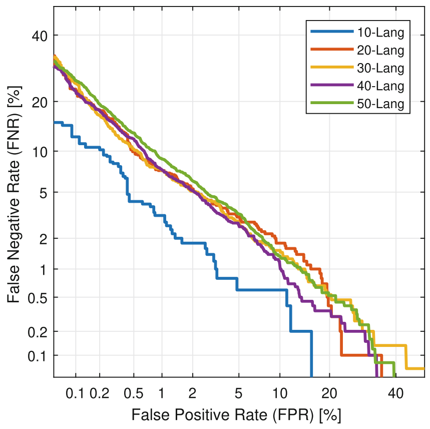


Fig. 3. DET curves for different sub-set target languages using DCNN.

Table 2 shows the EER when using the five sub-set target languages. As shown, when using the DCNN approach, the lowest EER is obtained, followed by SVM. The EER when using PLDA is the highest among the three classifiers. As the table shows, when DCNN is used, the EERs in the case of 20, 30, 40, and 50 languages are very similar. This result indicates the robustness of DCNN in different sub-set target languages.

Figures 1 and Fig. 2 show the DET curves in the case of 10 and 50 target languages, respectively. The figures make it clear that superior performance is obtained when the proposed deep CNN-based approach is used.

To investigate the effect of the training data size when using the three classifiers, an experiment is conducted using reduced training data. Table 3 shows the EER obtained in this case. The results show that using only 50 training i-vectors for each target language, the EER is still as low as 6.2%. As shown, the DCNN classifier shows the lowest EER, and it is followed by SVM. Figure 3 shows the DET curves for different sub-set target languages when using DCNN. As shown,

Table 4. Mean squared errors (MSE) for speech dereverberation.

$T_{[60]}$ [sec]	Denoising method		
	Reverberant	CNNDAE	DAE
0.30	1.64252	0.569070	0.570243
0.47	2.101042	0.569152	0.568758
0.60	2.346825	0.568656	0.569388
0.78	1.874870	0.568518	0.568780
1.30	1.701246	0.568803	0.572073

Table 5. Cosine similarities for speech dereverberation.

$T_{[60]}$ [sec]	Denoising method		
	Reverberant	CNNDAE	DAE
0.30	0.858569	0.960958	0.960942
0.47	0.789376	0.960941	0.960928
0.60	0.763129	0.960859	0.960889
0.78	0.824312	0.960880	0.960807
1.30	0.842408	0.961047	0.960879

differences can be obtained in the case where 10 target languages were used. In all other cases, the DET curves are very similar.

3.2 Front-End Feature Enhancement Experiments

This section presents the results when convolutional denoising autoencoder was used for feature enhancement. Twelve mel-frequency cepstral coefficients (MFCC) were extracted every 10 ms using a window of 25 ms. The MFCCs were then concatenated with shifted delta cepstra (SDC) coefficients to form feature vectors of size 112. The inputs to the convolutional denoising autoencoder (CNNDAE) are the feature vectors extracted from clean and reverberant speech, respectively. The encoder part of the CNNDAE consists of two convolutional layers and two *MaxPooling* layers. Each of the *MaxPooling* layers performs compression in the half-dimension. The decoder part consists of three convolutional layers and two *UpSampling* layers. The proposed method was also compared to a method based on a typical, fully connected denoising autoencoder (DAE) with one hidden layer and 32 units. Table 4 shows the MSEs for reverberant speech and dereverberant speech. As shown, in most of cases CNNDAE shows slightly lower MSEs compared to DAE. The MSE values show the effectiveness of using a deep convolutional denoising autoencoder in speech dereverberation. On the other hand, the results show that the performance when using CNN and DAE denoising autoencoders is closely comparable. Table 5 shows the cosine similarities. Similar to MSE values, when using CNNDAE the highest similarities are being obtained.

4 Conclusion

In this study, we proposed a method for language identification based on i-vectors and deep convolutional neural networks. The method was evaluated on the NIST 2015 LRE i-Vector Machine Learning Challenge task and demonstrated performance that is superior to that obtained using SVM and PLDA classifiers. For the identification of the 50 in-set languages, a 3.9% EER and a 16.0% cost were obtained. Using SVM, a 5.2% EER and 18.6% cost were achieved. Furthermore, a method for dereverberation in language identification was proposed. The proposed method is based on convolutional denoising autoencoder, and its effectiveness in speech dereverberation was demonstrated. As future work, spoken language identification experiments in reverberant environments based on a convolution denoising autoencoder will be conducted.

Acknowledgments. The authors would like to thank Professor Emeritus Péter Mihály, a former Professor of Eötvös Loránd University, Faculty of Humanities, Russian Department, Hungary, for fruitful discussions and his suggestions concerning current advances and future directions in spoken language identification.

References

1. Abdel-Hamid, O., Mohamed, A.R., Jiang, H.D., Deng, L., Penn, G., Yu, D.: Convolutional neural networks for speech recognition. IEEE/ACM Trans. Audio Speech Lang. Process. **22**, 1533–1545 (2014)
2. Caseiro, D., Trancoso, I.: Spoken language identification using the speechdat corpus. In: Proceedings of ICSLP 1998 (1998)
3. Cole, R., Inouye, J., Muthusamy, Y., Gopalakrishnan, M.: Language identification with neural networks: a feasibility study. In: Proceedings of IEEE Pacific Rim Conference, pp. 525–529 (1989)
4. Cristianini, N., S.-Taylor, J.: Support Vector Machines. Cambridge University Press, Cambridge (2000)
5. Dehak, N., A.T.-Carrasquillo, P., Reynolds, D., Dehak, R.: Language Recognition via Ivectors and Dimensionality Reduction. In: Proceedings of Interspeech, pp. 857–860 (2011)
6. Dehak, N., Kenny, P.J., Dehak, R., Dumouchel, P., Ouellet, P.: Front-end factor analysis for speaker verification. IEEE Trans. Audio Speech Lang. Process. **19**(4), 788–798 (2011)
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. 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
9. Ishii, T., Komiyama, H., Shinozaki, T., Horiuchi, Y., Kuroiwa, S.: Reverberant speech recognition based on denoising autoencoder. In: Proceedings of Intespeech, pp. 3512–3516 (2013)
10. Jiang, B., Song, Y., Wei, S., Liu, J.H., V.McLoughlin, I., Dai, L.R.: Deep bottleneck features for spoken language identification. PLos ONE **9**(7), 1–11 (2010)

11. 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)
12. 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)
13. L.-Moreno, I., G.-Dominguez, J., Plchot, O., Martinez, D., G.-Rodriguez, J., Moreno, P.: Automatic Language Identification Using Deep Neural Networks. In: Proceedings of ICASSP, pp. 5337–5341 (2014)
14. Lee, C.-H.: Principles of spoken language recognition. In: Benesty, J., Sondhi, M.M., Huang, Y.A. (eds.) Springer Handbook of Speech Processing. SH, pp. 785–796. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-49127-9_39
15. Leena, M., Rao, K.S., Yegnanarayana, B.: Neural network classifiers for language identification using phonotactic and prosodic features. In: Proceedings of Intelligent Sensing and Information Processing, pp. 404–408 (2005)
16. Li, H., Ma, B., Lee, K.A.: Spoken language recognition: from fundamentals to practice. Proc. IEEE **101**(5), 1136–1159 (2013)
17. 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)
18. Montavon, G.: Deep learning for spoken language identification. In: NIPS workshop on Deep Learning for Speech Recognition and Related Applications (2009)
19. Prince, S., Elder, J.: Probabilistic linear discriminant analysis for inferences about identity. In Proceedings of International Conference on Computer Vision, pp. 1–8 (2007)
20. Ranjan, S., Yu, C., Zhang, C., Kelly, F., Hansen, J.H.L.: Language recognition using deep neural networks with very limited training data. In: Proceedings of ICASSP, pp. 5830–5834 (2016)
21. Rawat, W., Wang, Z.: Deep convolutional neural networks for image classification: a comprehensive review. Neural Commun. **29**, 2352–2449 (2017)
22. Reynolds, D.A., Campbell, W.M., Shen, W., Singer, E.: Automatic language recognition via spectral and token based approaches. In: Benesty, J., Sondhi, M.M., Huang, Y.A. (eds.) Springer Handbook of Speech Processing. SH, pp. 811–824. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-49127-9_41
23. Shen, P., Lu, X., Liu, L., Kawai, H.: Local fisher discriminant analysis for spoken language identification. In: Proceedings of ICASSP, pp. 5825–5829 (2016)
24. Siniscalchi, S.M., Reed, J., Svendsen, T., Lee, C.H.: Universal attribute characterization of spoken languages for automatic spoken language recognition. Comput. Speech Lang. **27**, 209–227 (2013)
25. Nakamura, S., Hiyane, K., Asano, F., Endo, T.: Sound scene data collection in real acoustical environments. J. Acoust. Soc. Japan (E) 20, No. 3 (19995)
26. 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)
27. Zazo, R., L.-Diez, A., G.-Dominguez, J., Toledano, D.T., G.-Rodriguez, J.: Language identification in short utterances using long short-term memory (LSTM) recurrent neural networks. PLos ONE **11**(1), e0146917 (2016)
28. Zissman, M.A.: Comparison of four approaches to automatic language identification of telephone speech. lEEE Trans. Speech Audio Process. **4**(1), 31–44 (1996)

Arabic Named Entity Recognition Using Clustered Word Embedding

Caroline Sabty[1,2](✉), Mohamed Elmahdy[1], and Slim Abdennadher[1,2]

[1] Faculty of Media Engineering and Technology, German University in Cairo, Cairo, Egypt
{caroline.samy,mohamed.elmahdy,slim.abdennadher}@guc.edu.eg

[2] Faculty of Informatics and Computer Science, German International University, Cairo, Egypt

Abstract. Named Entity Recognition in Arabic is a challenging topic because of morphological and lexical richness of Arabic. In this paper, we propose an Arabic NER system that is based on word embedding. Word embedding hold semantic information about the context of the words. We hypothesized that the integration of word embedding features to the conventional lexical and contextual features could improve Arabic NER performance. The Conditional Random Field (CRF) sequence classifier was used. Since most CRF implementations only support categorical features, continuous word embedding vectors are clustered. In this paper, we are mainly investigating the effect of the number of clusters on NER performance. Moreover, the combination of fine and coarse grained clusters has resulted in further recognition improvement.

Keywords: Named entity recognition · Arabic · Conditional random field · Clustering · Word embedding · Natural language processing

1 Introduction

Arabic Natural language processing (NLP) has gained increasing importance and a lot of research has been conducted for various ranges of applications such as Name Entity Recognition (NER). It is the process of extracting and classifying the named entities from an unstructured text into a set of predefined classes [1].

One of the main statistical Machine learning techniques is Conditional Random Field (CRF). CRF is a sequence classifier to segment and label sequence data based on probabilistic models. It could be considered as an enhancement or generalization of Maximum Entropy (ME) and Hidden Markov Models (HMM) [2]. Recently, CRF has shown to be very successful in different NLP tasks and specially in NER [3]. One of the main advantages of CRF is the ability to consider contextual information before assigning a label to a word. N-gram algorithm and other available NLP techniques consider words as atomic units that do not have anything in common, which results in simplicity and capability to train big amount of data. However, these techniques recognize mostly words that are

A. Gelbukh (Ed.): CICLing 2018, LNCS 13396, pp. 41–49, 2023.
https://doi.org/10.1007/978-3-031-23793-5_4

available in the training data [4]. Moreover, [5] proved that using the concept of distributed representations of words like neural network based language models is better than N-gram models. The word representation (embedding) is a fixed size vector that capture the morphological and semantic information of the word.

Arabic is a morphologically rich language. As a lot of words have different prefixes and suffixes, which makes it hard to identify the similarity between them. However, with the usage of word embedding it is easy to find these similarities as similar words are mapped to nearby vectors. For instance, the words "الدولة" (the country), "الدولتان" (the two countries), "الدول" (the countries) differ in their morpholical forms, however, their embedding should be placed near to each other in the space. Arabic is also a lexically rich language, different words correspond to the same sense. For example, the two Arabic words "دولة" and "بلد" correspond to the same English meaning: "country".

One of the aims of this work is to show how unsupervised word embedding with CRF can be integrated to perform Arabic Named Entity Recognition task. This combination could not be done directly as CRF systems only allows categorical features and not continuous features such as word embedding vectors. The proposed solution is to cluster the generated vectors and plug the generated cluster IDs in the feature vector of the CRF system along with other lexical and contextual features. In order to cluster the generated vectors, we used K-means clustering. However, it is hard to know the optimal number of clusters. So, we compared different numbers of clusters to know which one gets the best performance. Our evaluation demonstrates the effectiveness of word embeddings clusters along with CRF for Arabic NER task.

In addition, some Arabic words can have different labels or NE types based on the context of the word. Thus, investigating the different contextual features for a given word might lead to a differently classified type. However, the question is how long should we consider in the left and right context of the word. A lot of studies have been conducted in this area and no final decision was taken, so we decided to evaluate different combinations to get the optimal context settings.

The paper is organized as follows. Section 2 presents some related work. In Sect. 3, the system design is illustrated by presenting the ANER pipeline, baseline features and word embeddings features. The evaluations and results are discussed in Sect. 4. Finally, Sect. 5 concludes the paper.

2 Related Work

Due to the morphological complexity of the Arabic language, few attempts have been done tackling the problem of NER in Arabic compared to other languages such as English. Two main approaches are being used in Arabic NER systems: the rule-based (RL) and machine learning (ML) based approaches [6]. The RL NER systems depends on grammar rules, usually represented as regular expressions. ML NER systems use learning algorithms that need large tagged training and testing data and use sets of features from these annotated datasets. Regarding the RL approach, it has been used by [7] for Arabic NER task. The system relied on a whitelist containing names and a set of grammar rules. It was evaluated using their own data set, the results of the evaluation accomplished high F1-measure of 85.9% for Location, 87.7% Person and 83.15 % for Organization. However in general, the RL approaches need a strong linguistic knowledge and they are time consuming.

Concerning the ML approaches, [8] used the Maximum Entropy (ME) and n-grams based algorithms for Arabic NER. They built a system (ANERsys) and they created corpus (ANERcorp) and gazetteer (ANERgazet) to training and testing. The system achieved 55.23% F1-measure. They enhanced the system in [9] by comparing two different techniques, Support Vector Machines (SVM) and CRF. In addition, they explored different combination of features such as contextual, lexical and morphological on different data sets. They could not state which one SVM or CRF is better as it differs for different entity types. The best results they achieved were 83.5% F1-measure for the ACE 2003 BN data. At the end in [3] they replaced the probabilistic model from ME to CRF and they got as a result 79.21% F1-measure. A set of features is being investigated in [10] to be used for CRF sequence labeling. They showed that character n-grams of leading and trailing characters of a word can be represented as a lexical features. This could help in the NER task without the use of linguistic analysis. They achieved 81% F1-measure on Benajiba dataset [8] and 76% F1-measure on ACE 2005 dataset. However, they have only considered three NE types (persons, locations and organizations).

Lately, a combination of both RL-based and ML-based approaches is used as a hybrid approach. For instance, in [11] they used a hybrid approach for Arabic NER. The RL part in their system is similar to the one presented in [7]. Regarding the ML part started by features and classifiers selections. Their approach showed promising results, however, it still has the problems of the RL approaches. The system performance was 90.1% F1-measure for Location, 94.4% for Person and 90.1% for Organization. Other available approaches has been used for Arabic NER such as using a leveraging parallel corpora (Arabic-Spanish) and previously developed tools for other language (Spanish) [12].

Few attempts have been done towards using word embedding for NER tasks in general. [13] Up to our knowledge, for Arabic language, only two systems used word embedding combined with CRF for Arabic NER. The first one is presented in [14], it showed promising results, however, it is designed to recognize entities extracted from social media text written in dialectal Arabic. It achieved 72.68% F1-measure on a dialectal dataset. In addition, few details are mentioned concerning the dimensions of the generated vectors and numbers of clusters. The second one presented in [15] mainly focused on comparing between two different word embedding algorithms (Word2Vec and Global Vectors). They used AQMAR Corpus which consists of 74k token. The best performance their system achieved was 67.22% F1-measure. Nevertheless, they did not study the effect of number of clusters.

3 Proposed System Design

The proposed ANER system implemented recognizes three different types of entities: Location (LOC), Person (PERS) and Organization (ORG). In addition, it tags entities that do not belong to these types with miscellaneous (MISC) and the ones that are not considered NE are tagged with other (O). The data used in the training and testing is the ANERCorp created by [8]. Labels follows the IOB format (classes) that is used in MUC-6 tasks[1]. Each class has two types: B-class and I-class. The B-class denotes the beginning of an entity and the I-class denotes the inside of a class. This data set is chosen because it is the largest free annotated corpus for ANER. It consists of 150,286 tokens and 32,114 labeled NE.

3.1 ANER Pipeline

As shown in Fig. 1, the system starts by normalizing the data. As the Arabic letters have different shapes, normalization process was needed to unify some letters that are written differently. For instance, "آ" and "أ" are replaced with "ا" and some punctuations such as '.' and ',' were removed. The word2Vec (W2V) model was trained using an independent Arabic newswire data-set. The normalized training data and the model were used to generate vectors for the training data. After, the vectors are clustered using a clustering model. The output from the previous step is used to generate IDs based on the cluster number for the training data. These IDs were added as features in the CRF system. Several features were added to the training data as will be explained later. In the final step, the new generated training data with all the features are fed as inputs to the CRF algorithm. The toolkit used to apply the CRF algorithm is CRF++ [16]. It is an open source CRF tool, used mainly for sequence classification. The main advantages of this tool is the ability to handle large feature sets and has the option of using multi-threading option which makes it much faster than other existing CRF toolkits.

3.2 Initial Baseline Features

Features are considered the characteristic attributes of words that should be used with ML algorithms [1]. The main selected features are: stemming, POS tagging, and some contextual and lexical features.

3.2.1 Stemming Features

In order to get the single representation of the words which is called Stem, a new approach should be implemented or used [17]. Adding the stems of the words as a feature to the CRF algorithm can help matching similar words with different morphological representations. Thus, we used the "ISRI stemmer" algorithm that is describes in [17] for word stemming on the whole dataset.

[1] http://cs.nyu.edu/cs/faculty/grishman/muc6.html.

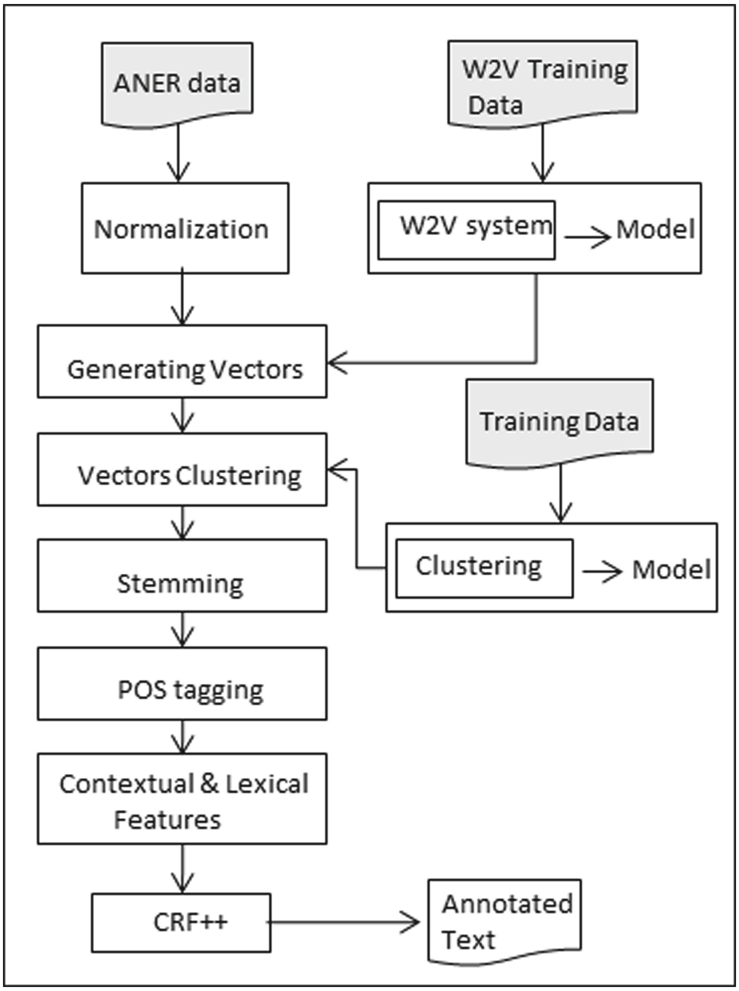


Fig. 1. A block diagram for the proposed system

3.2.2 Part-of-Speech Tag Features

Several POS tagging tools were investigated and the one that resulted in the best tagging performance is RDRPOSTagger. It is a language independent tool, that identifies part-of-speech tags (e.g., Nouns, Verbs). RDRPOSTagger applies an error-driven approach to construct a Single Classification Ripple Down Rules tree of transformation rules [18].

3.2.3 Contextual and Lexical Features

Contextual features are automatically generated based on the context of the NE. The context can be any number of previous or next words. We investigated several sets of features for this class of Contextual features to get the optimal settings. Concerning the lexical features, they are the character n-grams of the tokens, in other words, it could be considered as the fixed length prefix and suffix of a word. In our system the first and last letters of a word are added as two lexical features.

3.3 Word Embeddings Based Feature

In this work, Word2Vec is the algorithm used to produce the word embedding vectors. It is based on deep learning. And it uses skip-gram and Continuous Bag-of-Words (CBOW) models that is presented in [4]. CBOW predicts the probability of a word in a context. Whereas, skip-gram predicts the word based on a given context. W2V provides multiple degrees of similarity between different words by mapping to nearby vectors, which is very useful for Arabic language. A dataset is used to train the W2V model, that consists of 84M words.

The generated embeddings vectors are indirectly integrated in our system by adding them as a feature to the CRF algorithm to recognize Arabic NE. In order to convert continuous vectors into categorical features a clustering technique is used to cluster the vectors and adding their clustering IDs as a features. K-means is the clustering algorithm used. Two of the main factors that affect the performance of the system are the vector size and the number of clusters. The vector size chosen is 100. We have investigated several numbers of clusters as shown in Sect. 4.

4 Evaluation and Results

We divided the ANERCorp corpus into training and testing data of 110,286 and 40k tokens respectively. The evaluation process was divided into four experiments. First the best combination of contextual features was checked. Second, the performance of the system was evaluated by adding different features to build the baseline. The third experiment integrated the word embedding with the selected set of features and compared the different number of clusters for the word embedding. Finally, combining the coarse and fine grained clusters IDs was investigated.

4.1 Contextual Features

The first experiment started by evaluating the baseline with only the current word as a feature. Then, the next and the previous words were added to the current word separately. Furthermore, we have combined left and right contexts together. In the combination the left or right context consists of either one or two words. The performance measures used in our evaluation are precision (P), recall (R) and F-measure (F1).

In Table 1, the results of the different contextual features are listed. According to the Table 1, the best setting achieved by using the current and the previous word, they achieved 59.7% F1-measure. Whereas, the lowest results came from using only the current and the 2 next words 54% F1-measure.

4.2 Baseline Features Set

In Table 2, the performances of the CRF system using the different types of features are illustrated. The figure shows an increase in the value of the F1-measure by appending the features together. For instance, the addition of the lexical feature to the word stem and the current word increased the F1-measure from 64.1% to 66%. Moreover, by adding all the features to build the baseline F1-measure increased to 68.4%.

Table 1. The performance measures results in (%) from using different combinations of contextual features

Feature	P	R	F1
Current	97	41.7	58.3
Current-1Next	96.3	37.9	54.3
Current-2Next	95.5	35.3	51.5
Current-1Previous	98	43	59.7
Current-2Previous	97.1	40	56.6
Current-Previous-Next	96	40	56.4
Current-2Previous-2Next	96.9	37.5	54

Table 2. The performance measures results in (%) using different set of features

Feature	P	R	F1
Current-Stemming	96	48.2	64.1
Current-Stemming-Lexical	94.2	50.9	66
Current-Stemming-Lexical-Contextual	95.1	51.2	66.5
Current-Stemming-Lexical-Contextual-POStag	93.2	54.1	68.4

4.3 Cluster Granularity

In order to add the cluster ID of the word embedding as a feature, an experiment was done to get the relevant cluster granularity. Several number of clusters were experimented to get the cluster size with the best performance, the result of this experiment is shown in Table 3.

The evaluation started by adding the IDs of a small size cluster 50 to the set of features of the baseline. We kept increasing the size of the cluster and the performance increased as well till reaching the cluster with size 500 and after that the performance started to decrease. The figure indicates that the cluster with size 50 is the one with the worst results of 72.7% and the cluster with size 500 is the one with the best result of 76.1% F1-measure. Thus, adding the IDs of the cluster 500 to the feature set enhanced the performance of the baseline from 68.4% to 76.1%.

Furthermore, we investigated combining the coarse and fine grained clusters by adding the IDs generated by the number of clusters 50 and 500 to the feature set. The output demonstrated that the performance has slightly increased to 76.4% F1-measure. We interpret the improvement after the combination as coarse grained clustering could help more in modeling rare words. On the other hand, fine grained clustering can results in better performance for common words.

Table 3. The performance measures results in (%) using different number of clusters

No. of Clusters	P	R	F1
50	90	61.1	72.7
100	90	62.6	73.8
200	90.7	64.2	75.1
500	92	64.9	76.1
1,000	90.6	64.1	75
4,000	90.6	63.7	74.8
50 & 500 combined	91.4	65.7	76.4

5 Conclusion

An effective integration between CRF and word embedding for Arabic Named Entity Recognition task was presented in this paper. The benefits of using word embedding in Arabic NER task were illustrated. Word embedding can be integrated in CRF classifier by clustering the vectors and adding the cluster ID as a feature. Different number of clusters were investigated and 500 clusters achieved the highest performance, while 50 clusters got the lowest performance results. The best word embedding settings of combining fine and coarse cluster IDs results in 76.4% F-measure with a relative improvement from the baseline of 11.7%. Thus, we can conclude that the system achieved the best performance by the following set of features: Current word, Stemming, Lexical, Contextual, POS tagging, fine and coarse word embedding cluster IDs.

References

1. Nadeau, D., Sekine, S.: A survey of named entity recognition and classification. Lingvisticae Investigationes **30**, 3–26 (2007)
2. Lafferty, J., McCallum, A., Pereira, F.C.: Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: Proceedings of the 18th International Conference on Machine Learning, pp. 282–289 (2001)
3. Benajiba, Y., Rosso, P.: Arabic named entity recognition using conditional random fields. In: Proceedings of Workshop on HLT & NLP within the Arabic World, LREC, vol. 8, pp. 143–153 (2008)

4. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
5. Bengio, Y., Ducharme, R., Vincent, P., Jauvin, C.: A neural probabilistic language model. J. Mach. Learn. Res. **3**, 1137–1155 (2003)
6. Shaalan, K.: A survey of Arabic named entity recognition and classification. Comput. Linguist. **40**, 469–510 (2014)
7. Shaalan, K., Raza, H.: Nera: named entity recognition for Arabic. J. Am. Soc. Inf. Sci. **60**, 1652–1663 (2009)
8. Benajiba, Y., Rosso, P., Benedíruiz, J.: Anersys: an Arabic named entity recognition system based on maximum entropy. Computational Linguistics and Intelligent Text Processing, pp. 143–153 (2007)
9. Benajiba, Y., Diab, M., Rosso, P.: Arabic named entity recognition using optimized feature sets. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp. 284–293. Association for Computational Linguistics (2008)
10. Abdul-Hamid, A., Darwish, K.: Simplified feature set for Arabic named entity recognition. In: Proceedings of the 2010 Named Entities Workshop, pp. 110–115. Association for Computational Linguistics (2010)
11. Oudah, M., Shaalan, K.F.: A pipeline Arabic named entity recognition using a hybrid approach. In: Coling, pp. 2159–2176 (2012)
12. Samy, D., Moreno, A., Guirao, J.M.: A proposal for an Arabic named entity tagger leveraging a parallel corpus. In: International Conference RANLP, Borovets, Bulgaria, pp. 459–465 (2005)
13. Seok, M., Song, H.J., Park, C.Y., Kim, J.D., Kim, Y.S.: Named entity recognition using word embedding as a feature. Int. J. Softw. Eng. Appl. **10**, 93–104 (2016)
14. Zirikly, A., Diab, M.: Named entity recognition for Arabic social media. In: Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing, pp. 176–185 (2015)
15. Laachfoubi, N., et al.: Arabic named entity recognition using word representations. Int. J. Comput. Sci. Inf. Secur. **14**, 956 (2016)
16. Robert, P., David, G., Ke, C., Junbo, K., Kazuaki, M.: Arabic gigaword fourth edition ldc2009t30. Linguistic Data Consortium (LDC), Philadelphia (2009)
17. Taghva, K., Elkhoury, R., Coombs, J.: Arabic stemming without a root dictionary. In: International Conference on Information Technology: Coding and Computing, 2005. ITCC 2005, vol. 1, pp. 152–157. IEEE (2005)
18. Nguyen, D.Q., Nguyen, D.Q., Pham, D.D., Pham, S.B.: Rdrpostagger: A ripple down rules-based part-of-speech tagger. In: Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics, pp. 17–20 (2014)

Connecting the Content of Books to the Web and the Real World

Dan Cristea[1,2], Ionuț Pistol[2](✉), and Daniela Gîfu[1,2]

[1] Institute of Computer Science, Romanian Academy – Iași branch, Iași, Romania
daniela.gifu@info.uaic.ro

[2] Faculty of Computer Science, "Alexandru Ioan Cuza" University of Iași, Iași, Romania
ionut.pistol@info.uaic.ro

Abstract. In the last decade, the interest in developing a market of e-books has greatly increased, being preferred by many readers for their condensed volume, easy acquisition, some facilities that big providers started to introduce, such as the possibility to make personal notes, to search words in dictionaries and inside the book itself. However, this new format can have tremendously many more facets than proved by the printed book and has an extraordinary power to resonate differently with its particular readers. In this paper, we describe a technology that combines natural language processing with entity linking, web cartography, web mapping and augmented reality in order to bring to the reader of books new levels of knowledge, connection and leisure. The MappedBook is an electronic artifact that complements the textual content of a book with hyperlinks in a virtual space and augmented reality functionalities. Criteria to form social networks based on the MappingBooks technology are suggested. This study has a strong view on tourism business and educational systems.

Keywords: Reading · Name entity recognition · Entity linking · Geographical maps · Augmented reality · Evaluation of user satisfaction

1 Introduction

The last years witnessed a vividly developing market of e-books, which are preferred by many readers for their condensed volume, easy acquisition, some facilities that big providers started to introduce, such as the possibility to make personal notes, to search words in dictionaries and inside the book itself (e.g., the Amazon Kindle offer minimal, but very useful, interaction in the form of dictionary or Wikipedia definitions, lists of characters and their occurrences in the book) and, maybe not the least – ecological reasons, to diminish the abusive cutting down of trees. However, this new format can have tremendously many more facets than proved by the objects already existent in shops and implemented on reading devices.

The rich experience that a book creates for a reader is only partly due to the written text. Complementary knowledge adds to the genuine exposure of facts and descriptions. Moreover, the text has the extraordinary power to resonate differently in its particular

A. Gelbukh (Ed.): CICLing 2018, LNCS 13396, pp. 50–61, 2023.
https://doi.org/10.1007/978-3-031-23793-5_5

readers, and these inner evoked representations and feelings may or may not match what the author intended (which is, clearly, the same for all readers). Depending on the background of the reader, their preferences and hobbies, some mentions of the book may be ignored, some may be only shallowly deepened, and some may cause an interruption in the reading process to search for and retrieve additional information.

In this paper we describe a project and a technology intended to augment the text of a book with additional information acquired from the virtual and real world, which could be specific to the reader, as for instance their instantaneous position, the moment of reading and preferences.

The paper is organized as follows: Sect. 2 gives a general overview of a technology called MappingBooks, Sect. 3 describes the collection of data used to train the chain of automatic linguistic annotations, Sect. 4 presents the methodology that engines the automatic annotations, Sect. 5 shortly depicts the main features of the implementation, Sect. 6 discusses the evaluation of the technology, Sect. 7 prefigures other functionalities that regard social networking, Sect. 8 investigates connections of our work with other research, and Sect. 9 offers a free discussion on the benefits of this realization, with a glance into the future.

2 Architecture and Functionality

MappingBooks (MB) is a project developed between June 2014 and Sept. 2017, which developed a new type of electronic product with a high impact in education, and potentially on tourism and leisure. The specifications of the project state the main envisioned users of the developed MB application to be school pupils and students. The technology mixes methods from natural language processing, web cartography and mapping, mixed reality techniques and ambient intelligence/ubiquitous computing to link mentions of geographical entities existing in school manuals onto data existing on the web, to localize these entities on 2D hypermaps and to put them in correlation with the reader's location and related data.

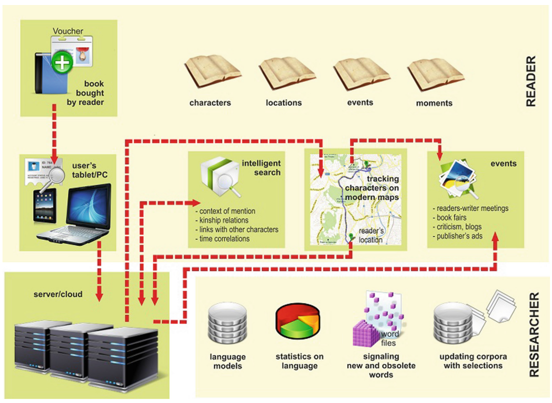


Fig. 1. A bird's eye view of the MB functionality

The main result of the MB project was a functional proof of concept of a *Mapped-Book*, i.e., an electronic artifact that complements the textual content of a book with hyperlinks in a virtual space and augmented reality functionalities.

Figure 1 shows the main ideas behind the project: when buying a book (in paper or eBook version) that is known to have a MappingBooks counterfact, the reader receives a voucher that allows them to install the application on their own mobile devices. The user's device will thus be coupled with a server which hosts the linguistically and geographically processed book and will allow two main types of interactions between user and app: either at the user's initiative, allowing them to search the electronic text and connect to sources of information related to the text on the web, or at the app's initiative, being prompted when the user happens to be physically located in the proximity of geographical entities mentioned in the book. Moreover, the dense annotations automatically produced over the texts and their links in the internet could be accumulated on the server and offered freely for computational linguistics research purposes.

For the linguistic processing, the project implements an NLP chain that includes: name entity recognition (NER-phase 1), sentence spitting, tokenization, POS-tagging, noun phrase (NP) chunking, NER-phase 2 and semantic relations detection. In MB, like in TTL (Tokenizing, Tagging and Lemmatizing) (Ion, 2007), some NER functions are applied before the sentence splitter, in order to correctly recognize dots to abbreviation as opposed to those signaling limits of sentences. More evolved NER functions are applied after the NPs are identified. Our NER implementations mix brute force methods (as provided by the use of a large collection of proper nouns, including abbreviations) with symbolic methods (a collection of regular expressions used to disambiguate types of proper names as indicated by contexts). Once name entities of type location are identified, semantic relations are searched for, by applying a collection of lexical-syntactic patterns anchored in triggers (keywords specific to different types of semantic relations).

Further on, mentions of geographic locations in the original text of the book are linked into the virtual space (in this implementation – Wikipedia). The application is sensitive to the location where the user (presumably the book reader) happens to be while using the application, as seized through their mobile phone or tablet, and if the mobile device is a tablet, elements of augmented reality can be experienced, as for instance arrows superposed with the captured image that indicate locations mentioned in the book and which are in a reachable vicinity of the reader. The project demonstrated these ideas by elaborating a functional prototype and transposing into the MB technology a first book, an 8th grade Geography schoolbook in Romanian[1].

3 Data Set

Texts are made of mentions of entities, events, thoughts, sensations, placed on one or more temporal axis. Representing these elements belonging to a more or less concrete world and recovering in our minds connections between them is indispensable for understanding the text. Without them, we would not be able to express any continuous thought rooted in the text, perhaps more than remembering a succession of places, characters, sensations and ideas, but each isolated from the others.

[1] Neguț, S., Apostol, G., Ielenicz, M.: "Geografie", Humanitas Educațional, București (2008).

In MB, references to geographic entities are one or more contiguous words (usually NPs) that identify real-world locations (e.g., *Romania*, *Black See*, *mount Ceahlău*). The notion of a geographical entity is a customization of the more general notion of named entity, which identifies textual terms whose meanings are known outside the text in which they are mentioned. In the case of geographical entities, external significance has the particular property that it can be generally associated with an area of a map and it possesses some specific properties (height, surface, population, etc.).

To help the machine learning process, we have manually developed an annotated file showing in XML entities and relations between them at both the surface language level and the knowledge representation level (Sălăvăstru and Gîfu 2015; Gîfu et al. 2015). Although, in general, entities mentioned in a text are of a very diverse nature (persons, animals, places, organizations, moments of time, etc.) in this study we have been concerned only on persons and locations. Name entities (ENTITY elements) include at least one WORD and contain minimum the TYPE and SUBTYPE attributes. We exemplify below the main types and subtypes.

a) Location (TYPE = “PLACE”), with subtypes: “VILLAGE”, “TOWN”, “CITY”, “DISTRICT” and “REGION”.
b) Relief form (TYPE = “LANDFORM”) with subtypes: “FIELD”, “PLATEAU”, “HILL”, “PEAK”, “BOTTOM-LAND”, “DEPRESSION”, “DEFILE”, etc.
c) Water (TYPE = “WATER”) with subtypes: “LAKE”, “BROOK”, “DELTA”, etc.
d) Size (TYPE = “DIMENSION”), naming lengths, elevations of mountains, distances between localities, depth of the waters, etc. Subtypes here are: “HEIGHT”, “DEPTH”, “LENGTH”, etc. with the unit of measurement expressed by the MEASUREMENT attribute (“M” for meter).

Among the extremely large class of semantic relations that a text could express, we deciphered only:

a) “REFERENTIAL”, with subtype “COREF”, to express coreferential links, also complemented with the attributes FROM (indicating the ID of the right-most-in-the-text entity) and TO (its antecedent). For instance:
   1: [*Romania*]… 2: [*our country*]… ⇒ < RELATION TYPE = “REFERENTIAL” SUBTYPE = “COREF” FROM = “2” TO = “1” / >;
b) Spatial relations: TYPE = “SPATIAL”. Examples of subtypes are:

   – "NEAR” and “FAR” – to express the proximity and farness between two locations,
   – "POSITION”, with the cardinal point indicated by the CARDINAL argument, for instance:

1: [*Parâng Mountains*]… *is located at the East of* 2: [*Retezat Mountains*] ⇒ < RELATION TYPE = “SPATIAL” SUBTYPE = “POSITION” CARDINAL = “EST” FROM = “1” TO = “2”.

– "PART-OF”, for instance:

1: [*Romania*] *concentrates on its territory two thirds of the chain of* 2: [*Carpathian Mountains*] ⇒ < RELATION TYPE = "STRUCTURAL" SUBTYPE = "PART-OF" FROM = "2" TO = "1".

– "SOURCE", indicating the source of a running water, as here:
  1: [*Târnava Mică*] *with springs in* 2: [*Gurghiu Mountains*] ⇒ < RELATION TYPE = "STRUCTURAL" SUBTYPE = "SOURCE" FROM = "1" TO = "2".
– "CONFLUENT-OF", indicating the confluent of a river, as here:
  The basin of 1: [*Mureş*]… 2: [*Târnava*], *its most important confluent* ⇒ < RELATION TYPE = "STRUCTURAL" SUBTYPE = "CONFLUENT-OF" FROM = "2" TO = "1".

In the present implementation we were not concerned with: negative relations (*Romania… is not part of the Balkan Peninsula*), spatial relations of a very complex nature (*Romania is at a latitude halfway between the Equator and the North Pole*), or past names of toponyms (*Histria*, *Tomis*, *Callatis*, *Apulum*, *Ampelum*, *Napoca*, *Potaissa*, *Sucidava*, etc.).

4 Methodology of Generation of Inner and External Links

The main module of our NER is a gazetteer (GAZ), which uses a list of toponyms and other geographical names grouped by categories in order to identify potential candidates. Given the nature of the texts actually used to develop the MappingBooks project (all in the domain of Romanian geography schoolbooks), the gazetteer was developed to include as many geographic names as relevant to the geography of Romania. We identified 15 general types, including 103 subtypes, out of which 67 are of a geographic nature: entities of these types may be associated with locations on the map. In order to populate the gazetteer based on the defined types, we used Geonames as the main resource, frequently used by the community of researchers and developers in the field. Geonames has been developed from free government and educational resources, later supplemented with user-contributed data and validated by specialists. For Romania, Geonames includes 25,951 unique entities, with over 45,000 alternative names.

To complete the Geonames geotagger, it was necessary to map the types / subtypes defined in MappingBooks to those in Geonames. Due to the much larger list of subtypes in Geonames (652 vs. 67 in MappingBooks), we generally paired multiple Geonames subtypes to one MB subtype. The lower number of subtypes in our classification reduces the ambiguity of the GAZ module, the level of detail being considered sufficient for a potential user of the application (Cristea et al. 2016a).

In parallel, a pattern-matching module (PAT) uses a set of templates, identified in that part of the text manually annotated, to find other candidates for geographic entities. The major difference between GAZ and PAT modules is that GAZ uses the exact textual form of potential candidates, while PAT also uses contextual information.

The templates used have been built using Graphical Grammar Studio (GGS) (Simionescu, 2011), a graphical environment that allows the development and application of contextual grammars on annotated and raw texts. GGS defines a constraint language that allows the description of composite elements, complemented with look-ahead and look-behind assumptions, and the association of scores on the arcs defining the parsing sequence. A network described in GGS (directed chart) consumes an input and produces an annotated output with additional information according to the paths traversed in the network. The nodes in these graphs define conditions that can be reached, while the directed arcs identify conditions for consumption of the input tokens. A GGS grammar is basically a finite state machine. The PAT module receives an XML annotated document, resulted from the sentence splitting, tokenization and POS-tagging pipeline, and a GGS network. If, corresponding to a subsequence of the input sentence, one path is found that links the initial node to the final node in the network, then that input sequence is consumed and an XML sequence is generated. The result is a document containing annotations added over the names found in the resource used (type, subtype, coordinates, other geographic attributes). In the case of an ambiguity (identifying the same name in different categories), the annotation will add all possible variants and the decision is taken by the PAT module.

In (Cristea et al. 2015), the issue of annotating relations linking entities in texts is addressed. In the mentioned paper we say that any mention of an entity (restricted only to persons, gods, group of persons and of gods) is a mapping from a text expression to a corresponding «container», which is associated with each entity at its first mention and is subsequently filled in with pieces of information that define details, as contributed by the text (name, sex, kinship and other semantic relations, etc.). In all corpus-based approaches, mentions, not containers, are annotated, but if semantic reasoning is tried on these annotations, then the more complex containers are the keepers of semantic representations. In MappingBooks we adopt the same simplified representation, associating inner links to mentions of entities and not to their containers. As such, all relations will have a local dimension, being perceived contextually, in the exact place of their occurrence in the text.

The last step of the linguistic processing chain is realized by the Relations Detection module (RD), which is responsible for putting in evidence semantic relations mentioned in the text. The set of patterns of the RD module are regular expressions (Colhon et al. 2016) that link fixed textual sequences (for instance: *is/was redone by*, *is/was finished by*, *is/was the creation of*, *is/was realized by*, etc.) with two slots, XML ENTITY elements (in the example above, the first of type institution and the second of type person). This way, to take one example, buildings built by persons can be evidenced in the text.

Other relations put in evidence geographical relations that link two entities of type location. All patterns are extracted from examples, by observing common types of relevant text fragments. At the end of the described chain, a heavily annotated XML file is produced. Thus, the entity names, as identified by the NER component, are enriched with information of historical and geographical nature.

5 Interface

The user interface scales down to different users' devices running Android. Without entering into details, the dimension of each displayed component is expressed in percentages of the dimensions of the screen. Thus, on a larger screen device, a button will appear larger, and on a smaller device, it will appear smaller. In other words, the same aspect ratio will be maintained regardless of the device. This solution involves also the use of different geographical resources of different pixel densities.

The following geographical resources are running on the server, continuously fetching data towards users' devices: GeoNetwork[2] – an open source web platform allowing the creation of spatial data catalogues, the editing of their metadata and spatial search by key words; OpenLayers 3.19.1[3] – a Javascript library that allows the display of web maps in any browser; Nominatim – a search engine for geo-data operated by a web interface; GeoServer – a Java-based application that implements web services to create and deliver spatial data as maps.

Finally, an augmented reality application (Cristea and Pentiuc 2016) couples the geographical position of the device with the image captured by the built-in camera, on which it projects Points Of Interest (POIs) that appear mentioned in the book and are momentary in the close vicinity of the user. Different pruning strategies are used to cut down the number of POIs when an area larger than 5 km is selected by the user, or otherwise the image would become too dense in information and hard to read. The purger ranks the localities in the selected area by the population number (information stored on the geo-server) and retains the most important. Other strategies have been used to stabilize the image (for instance, a frequency actualization threshold had to be established for purging of the data coming from the built-in compass, originally too sensible at very small rotation angles). When the user touches a POI displayed on the image, the interface will display its mentions in the text as well as supplementary information got from external sources (Wikipedia).

6 Results

A development objective of the mobile application created within the MB project has always been the users' satisfaction. The target group identified for the purpose of demonstrating the application was formed of school pupils studying "Geography".

After several stages of application testing in partners' project laboratories, we went to evaluate the app in the context of the identified target groups. The investigative approach trying to understand how potential users relate to MappingBooks was based on the use of a social survey using the questionnaire as a working tool. The investigation was based on twenty closed statements, with Likert type scales, and two open questions (Lewis and Sauro 2009), and mainly pursued:

- measure the overall satisfaction (usability);
- measure the change of appetite (negative-positive) of pupils towards the studied notions when using the MB technology as compared to the classical use of the manual;
- identify the interestingness of the implemented features of the application;
- identify advantages and disadvantages;
- identify design suggestions (recommendations).

The sample used for the assessment was probabilistic, consisting of 156 respondents, VIIIth and IXth grade school pupils from two Romanian cities (Iași and Botoșani). The construction of the sample was that of the "snowball" (Sauro and Lewis 2012). The

[2] https://sourceforge.net/projects/geonetwork/

[3] https://openlayers.org

sample has the following characteristics: 50% male, 50% female; 21.2% were in the VIIIth grade, and 78.8% – in the IXth grade; of those in the IXth grade, 59.3% are in the mathematics-computer science profile, 15.4% in the philology profile, and 25.2% in the natural sciences profile. In relation to the general connection of the respondents with the technologies, 53.2% use the mobile device very frequently, 30.8% – frequently, 10.3% – rarely and 5.8% – very rarely or never.

The possible answers were scaled from the lowest to the highest values of the item "agreement". 60.3% of respondents agreed with the statement "I like to use this app" and 30.8% were undecided. 90% of questioned people agreed that the app could be potentially further developed. Significant is also the score of over 80% for agreeing to the statement "The application helps me understand the notions of geography better than the schoolbook", which indicates the direct didactic benefits perceived by the target group in fixing the notions expressed in the text.

The lowest total agreement (58.3%) was received for the statement, "I like how I can interact with the text of the book in the application." The dissatisfaction proved to be related to the type of the device: the majority of pupils were having mobile phones and therefore small size screens, as compared to larger displaying texts on tablet screens. Actually, 66% of the respondents agreed with the statement "I would like the mobile device we tested the app on to have a larger screen."

A significant difference in satisfaction was between the VIIIth and IXth graders, the IXth graders proving more than 20% greater agreement on the statement "The application is easy to use", respectively 48.8% versus 27.3%. The probable cause of this difference is the greater experience that class IX students have with both the contents of the manual and the use of test devices.

Regarding the possible enhancements to the application (included in the last 7 statements of the questionnaire), the highest value of the "agreement" item was obtained for "being able to communicate with other users via the application" (23.7%) (Novak-Marcincin et al. 2014; Gifu and Teodorescu, 2014), a recommendation that we retain for possible future developments (see the next section for a short description).

7 Networking Rooted on Common Readings

The evaluation seems to reflect a strong desire of users to interact with other people through apps like MB. As proved in those areas in which NLP technologies have been particularly successful in enhancing a social network user experience, which are e-learning (Romero and Ventura, 2013) and consumer reviews (Netzer et al. 2012), links between users can be established based on their consumed product, in both cases this being shown to significantly increase users' satisfaction and retention.

When the processed text (the host document) will be freed from the constraints of approaching only the domain of teaching of geography, which we had to observe in the present deployment, the annotations automatically added to the text could be extremely rich. Items marked could be entities with very diverse real-world significance: locations, institutions, notorious people, historical events, etc. as well as links between these entities, both inside and outside the text. But semantic and locational links could be a good source of forming social communities of readers. Some examples of possible social networks are (Cristea et al. 2016b):

- co-readers of a certain book B could be formed out of those readers who have declared the "subscripted for B" field visible;
- co-readers of a certain book B AND actually being in the proximity of a location L would include readers who also declared visible their "instantaneous location";
- co-readers of B (a book) AND co-track of T (a trajectory rooted in that book) would include readers reading a book and having followed a more or less similar sequence of places mentioned in that book (as appears, for instance, in a traveling guide).

It is easy to imagine other ways to form social communities rooted in readings. Imagine, for instance, that common readings would be intersected with attended places in the past and levels of friendship reported by social media, like Facebook of WhatsApp, or real-world events and entities mentioned in a book associated with real-world locations and particular moments of the year/day.

A good example of maps rooted on literature are those built within the Vilnius' literature mapping project[4], developed by several Vilnius University literature scholars and cartographers. Their maps, with data manually acquired from Lithuanian texts, include mentions of places of the city of Vilnius. It is clear that the MB technology could be used to automatize such an enterprise, to build cultural maps and to connect people based on their readings. In enterprises of this kind, a great importance will have name entity coreferences in multiple documents.

8 Related Work

While performing this study, we had in mind three components, for which we searched appropriate approaches in the literature: natural language processing (NLP), linguistic linked open data (LLOD) and information extraction (IE).

As an NLP approach, entity identification and linking touches some issues related to word sense disambiguation. Wikipedia and Open Linked Data entities inventories combine numerous sources of knowledge, such as DBpedia, WordNet, Freebase, GeoNames, US Census, PubMed, etc. The task of knowledge base population with a subtask of entity linking was launched in 2009 (McNamee and Dang, 2009) focusing the research in this area and providing gold standard datasets and evaluation measures. This task is still very challenging; the most successful approaches rely on ranking the candidates for linking in a supervised manner, using a diversity of features like similarity metrics, entity profiling and so on. To exploit the coherence of a text as a "feature", some methods can target single entities at a time (Zheng et al. 2010) and other several groups (Milne and Witten 2008; Wick et al. 2013; Cheng and Roth 2013). It is well known that the performance of the systems depends on the evaluation metrics in connection with how a task is formulated. State-of-the-art results in named entity recognition and linking reach an F-score of approximately 77% (Ji et al. 2014). In MB, we restricted the task of entity identification and linking to only full and pronominal mentions, by applying a rudimentary anaphora resolution engine, ignoring other types of surface referencing. Zhou et al. (2014) report successful supervised entity discovery and linking, but our task was much simplified by the access to collections of names, such as Geonames.

[4] http://www.vilniusliterature.flf.vu.lt/en/?page_id=18.

Relations detection, as performed in MappingBooks, has many similarities with task of Information Extraction (IE) – for instance, detection of the date of an event. In MB the goal of applying IE techniques was to gather additional information that could be useful to the reader. Classical exercises campaigns in IE are the DARPA initiated Message Understanding Conference series, which covered various types of extracted information – from structured relations, to more free-form text fragments. Approaches range from unsupervised (Riedel et al. 2010) or Open Information Extraction (Wu and Weld 2010) to self-supervised (Downey et al. 2010; Fader et al. 2011) and supervised and targeted using seed relations from resources such as FreeBase (Mintz et al. 2009) or Wikipedia infoboxes (Wu and Weld 2010).

9 Conclusions

It is clear that not all types of books should stand the same MappingBooks processing in order to become a *MappedBook*. A large palette of editing tools can be put in the hands of the book editor that they could employ to augment the leisure and the benefits of the reading act.

Certain semantic processing techniques – such as named entity recognition and entity linking – are largely insensitive to genres and topics. Others, like event identification, temporal analysis and coreference resolution, may be more difficult in the presence of dialogue and shifting or alternating timelines. Fiction and non-fiction books may have different requirements with respect to information to be added to the identified entities, and within fiction books we may encounter a mixture of fictional and non-fictional characters, events and locations. The purpose of the final eBook object could also vary – being destined for education at various levels, as guides during travelling, or reading for pleasure.

Our vision is that the MappingBooks technology, or one emerged from ideas developed in this project, could be put at the basis of a future printing house technology, in which the book editor would be given a palette of tools, any of them added to the book under processing, yielding the inclusion in the final electronic artifact of a theme that exploits the linkage of entities, internally or/and externally, including elements of augmented reality or not, synchronized or not to the specificities of the reader, able to trigger or not formation of social communities, etc.

All these being set, we are also aware that a technology like MB should be handled with care. One cannot avoid annoying questions, like the following: to what degree should the technology interfere with the act of literary consumption? Augmented reality techniques are at our touch, still which of the descriptions of perception mentioned in a book should be added to the text? For instance, perhaps we would not want to rebuild in the virtual space sensations, images, sounds or smells as described by the author of a fiction book. What we want is only to bring the reader to another level of knowing notorious entities mentioned there and maybe to another level of sensing the text. But how rich in electronic effects should this level be must be judged well or otherwise the reader can be suffocated and the text trivialized.

Acknowledgments. The paper as published in this volume has only minor changes with respect to the version accepted in the 2018 CICLing Conference. MappingBooks was financed by UEFISCDI

(Executive Agency for Higher Education, Research, Development and Innovation Funding) of the Romanian Ministry of Education and Research, under the Partnerships Programme (PN II, competition PCCA 2013), project code: PN-II-PT-PCCA-2013–4-1878. Partners in the project have been: „Alexandru Ioan Cuza" University of Iași, SIVECO Romania, and „Ștefan Cel Mare" University of Suceava. Our colleagues Mihai Niculiță and Dorina Ticu have designed and implemented the geographical mapping and the evaluation with subjects, respectively. We thank Tommaso Caselli, Piek Vossen, Vivi Năstase, Danilo Giampiccolo and Carlo Strapparava for a past collaboration fostering some ideas mentioned in this paper. Part of the research described in this paper has been performed within the research plan of the Institute of Computer Science of the Iași branch of the Romanian Academy.

References

Cheng, X., Roth, D.: Relational inference for wikification. Proceedings of EMNLP (2013)

Colhon, M., Cristea, D., Gîfu, D.: Discovering semantic relations within nominals. In: Trandabăț, D., Gîfu, D. (eds.): Proceedings of the Workshop on Social Media and the Web of Linked Data, RUMOUR-2015, A satellite event of EUROLAN-2015, Sibiu, Romania, July 2015, Springer (2016). https://doi.org/10.1007/978-3-319-32942-0_6

Cristea, D., et al.: Quo Vadis: A corpus of entities and relations. In: Gala, N., Rapp, R., Bel-Enguix, G. (eds.) Language Production, Cognition, and the Lexicon. TSLT, vol. 48, pp. 505–543. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-08043-7_28

Cristea, D., Pentiuc G.-Ș.: A new ebook concept and technology dedicated to geographical information. In: Proceedings of the 18-th International Conference on Scientific Research & Education in the Air Force – AFASES, Brașov, pp. 417–412 (2016). https://doi.org/10.19062/2247-3173.2016.18.1.57

Cristea, D., Pistol, I., Gîfu, D., Anechitei, D.: Networking readers: using semantic and geographical links to enhance e-books reading experience. In: Nguyen, N.-T., Manolopoulos, Y., Iliadis, L., Trawiński, B. (eds.) ICCCI 2016. LNCS (LNAI), vol. 9876, pp. 59–67. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-45246-3_6

Downey, D., Etzioni, O., Soderland, S.: Analysis of a probabilistic model of redundancy in unsupervised information extraction. Artif. Intell. **174**(11), 726–748 (2010)

Fader, A., Soderland, S., Etzioni, O.: Identifying relations for open information extraction. Proceedings of EMNLP (2011)

Gîfu, D., Teodorescu, M.: Communication concepts vs. sciences concepts. In: ILSHS, **18**, pp. 48-57 (2014)

Gîfu, D., Pistol, I., Cristea, D.: Corpus of entities and semantic relations with application in geographical domains. In: Proceedings of the 11th ConsILR, Gîfu, D., Trandabăț, D., Cristea, D., Tufiș, D. (eds.): "Alexandru Ioan Cuza" University Publishing House, Iaşi, pp. 67–78 (2015)

Sălăvăstru, A., Gîfu, D.: Annotating geographical entities. in: research in computing science. Al. Gelbukh, Samhaa R. El-Beltagy (eds.), issue 90: Advances in Computational Linguistics, México, pp. 45–56 (2015)

Ion, R.: Methods of automatic semantic disambiguation. applications for english and romanian (in Romanian). PhD. Thesis, Research Institute for Artificial Intelligence, Romanian Academy, Bucharest (2007)

Ji, H., Nothman, J., Hachey, B.: Overview of the TAC-KBP 2014 Entity discovery and linking tasks. In: Proc. Text Analysis Conference (TAC) (2014)

Lewis, J.R., Sauro, J.: The factor structure of the system usability scale. HCII 2009, San Diego CA, USA (2009)

McNamee, P., Dang, H.T.: Overview of the TAC 2009 knowledge base population track. In: Proceedings Text Analysis Conference (TAC) (2009)

Milne, D., Witten, I.: Learning to link with Wikipedia. In: 17th ACM Conference on Information and Knowledge Management, Napa Valley, CA, pp. 509–518 (2008)

Mintz, M., Bills, S., Snow, R., Jurafsky, D.: Distant supervision for relation extraction without labeled data. In: Proceedings of ACL (2009)

Netzer, O., Feldman, R., Goldenberg, J., Fresko, M.: Mine your own business: market structure surveillance through text mining. Mark. Sci. **31**(3), 521–543 (2012)

Novak-Marcincin, J., Gîfu, D., Nicolescu, A.: The standard of axes in ontology of communication. In: International Letters of Social and Humanistic Sciences **30**(2), 176–183 (2014)

Riedel, S., Yao, L., McCallum, A.: Modelling relations and their mentions without labelled text. In: Balcázar, J.L., et al. (Eds.): ECML PKDD 2010, Part III, LNAI 6323, pp. 148–163, Springer-Verlag Berlin Heidelberg (2010). https://doi.org/10.1007/978-3-642-15939-8_10

Romero, C., Ventura, S.: Data mining in education. Wiley Interdisc. Rev. Data Min. Knowl. Disc. **3**(1), 12–27 (2013)

Sauro, J., Lewis, J.R.: Quantifying the User Experience: Practical Statistics for User Research. Morgan Kaufmann, Waltham MA, USA (2012)

Simionescu, R.: Graphical grammar studio as a constraint grammar solution for part of speech tagger. In: Proceedings of the International Conference Resources and Tools for Romanian Language – ConsILR-2011, Bucharest, "Alexandru Ioan Cuza" University of Iași Publishing House (2011)

Wick, M., Singh, S., Pandya, H., McCallum, A.: A joint model for discovering and linking entities. In: Proceedings of the Third International Workshop on Automated Knowledge Base Construction (AKBC) (2013)

Wu, F., Weld, : Open Information Extraction using Wikipedia. Proceedings of ACL (2010)

Zheng, Z., Li, F., Huang, M., Zhu, X.: Learning to link entities with knowledge bases. In: Proceedings of NAACL-HLT (2010)

Zhou, S., Kruengkrai, C., Inui, K.: Tohoku's entity discovery and linking system at TAC 2014. In: Prof. Text Analysis Conference (TAC2014)

Finding Questions in Medical Forum Posts Using Sequence Labeling Approach

Adrianus Saga Ekakristi, Rahmad Mahendra(✉), and Mirna Adriani

Faculty of Computer Science, Universitas Indonesia, Depok, Indonesia
`adrianus.saga@ui.ac.id`, `{rahmad.mahendra,mirna}@cs.ui.ac.id`

Abstract. Complex medical question answering system in medical domain receives a question in form of long text that need to be decomposed before further processing. This research propose sequence labeling approach to decompose that complex question. Two main tasks in segmenting complex question sentence are detecting sentence boundary with its type, and recognizing word that could be ignored in sentence. The proposed sequence labeling method achieves F1 score of 0.83 in detecting beginning sentence boundary and 0.93 when determining sentence type. When recognizing the word sequence that could be ignored in sentence, the sequence labeling method achieves F1 score of 0.90.

Keywords: Medical question answering · Question decomposition · Sequence labeling · Chunking

1 Introduction

Common factoid question answering system receives a one simple question that could be answered with simple fact. This system cannot answer health-related question from society in general. When consulting with a doctor, people tend to describe their symptom in multiple sentences and then ask one or more question. Most of them even say greetings or thank you. The complex question can be decomposed into several parts, i.e., contextual information about patient's disease, question, or sentence that can be ignored. Consider the following example:

- umur saya 21 tahun. sudah 4 hari ini panas badan saya turun naik, sebenarnya saya sakit apa??? terima kasih.
  `(my age is 21 years old. for these past 4 days, my body heat was getting up and down, what is actually my disease??? thank you.)`
- Lalu bagaimana cara pengobatannya ya, dok?
  `(And how to cure it, doc?)`

The first example of complex question above consists of multiple sentence background information, followed by a question, and closed with a gratitude sentence. The last sentence can be ignored because it doesn't contain useful

A. Gelbukh (Ed.): CICLing 2018, LNCS 13396, pp. 62–73, 2023.
https://doi.org/10.1007/978-3-031-23793-5_6

information about patient's symptom or disease. We can also observe the use of informal grammar and structure, such as using comma at the end of the sentence.

In the second example, we can see that the user use greeting word "dok" that refer to the doctor. Greeting word like this does not contain any useful contextual information about patient's symptom and could be ignored. A question answering system should first be able to recognize various sentences, whether a question, a description of user's symptom, or other parts that should be ignored.

This process of decomposing complex question text is a crucial step in medical complex question answering system. In this paper, we propose sequence labeling approach for decomposing complex question text. The module we build receives a consultation request text as an input and return a list of sentences with its type as an output. The data we used for training and testing is question and answer pairs from health consultation forum.

2 Related Work

Even though the rule-based method sometimes gives high accuracy in segmenting text into sentences, the use of sequence labeling approach is able to enhance the model performance. It is not only proved to increase the accuracy, but also easier to maintain than hand-crafted rules and mitigate the language-specific restriction of the method. Evang et al. [2] defined a set of IOB tag that represent the beginning of a sentence, beginning of token (word), inside a token, and outside the token. Conditional Random Field was chosen as the model and character-level word embedding as the feature. The proposed method was then tested in three news corpus, each has different languages. The result show higher F1 sentence score than state-of-the-art boundary detection system.

Several works have applied different NLP techniques to decompose question text into atomic question along with contextual background information. Each of those atomic question could then be answered by question answering system. Roberts et al. [6] proposed a system consists of several module to decompose English consumer-health complex question. A question text, which they called request, was segmented and classified into several sentence type, which are question, ignore, and background. The question sentence would undergo several decomposition process, which are decomposing clause-level question, decomposing phrase that spans a set of decomposable items, and decomposing phrase that spans an optional item. Furthermore, Roberts et al. [6] also proposed classification for sub-class of background sentence and focus recognition based on detailed annotation [5]. Their proposed methods for those modules were mostly combination of rule-based candidate generator followed by rank-and-filter machine learning methods. The tokenization and sentence segmentation process were not described in detail, most likely because rule-based method already produced a good result for relatively good-grammar questions.

Sondhi et al. [8] on the other hand, proposed a method to extract medical problem and medical treatment descriptions from medical forum data. The data itself were taken from HealthBoards[1] and manually annotated. Support Vector

[1] https://www.healthboards.com/.

Machine and Conditional Random Field is chosen as the model and various features were proposed, such as n-gram language model, the Unified Medical Language System (UMLS)[2] semantic groups of words, position features, heuristic user-based features, previous sentence tag, length-based features, and various morphological features. The proposed method achieve best prediction accuracy above 75%.

For Indonesian language, Hakim et al. [3] have built a corpus consisting of medical questions taken from five different health consultation website. The corpus comprises 86731 question-answer pairs which are produced by the interaction of patient and doctor in the forum. Mahendra et al. [4], on the other hand, proposed a method for decomposing Indonesian question text in the corpus. The three main components are sentence splitter, sentence classifier, and multi-questions splitter. Sentence splitter segmented the question paragraph into a list of sentence using rules, such as delimiters and heuristic rules. In sentence classifier component, Support Vector Machine model was used to classify sentences into types defined by Roberts et al. [6]. The features used are n-grams, position and length of sentence, question-specific attributes, and dictionary of symptoms and diseases. For multi-questions splitter, several strategy was proposed to accommodate different combination of sentence types in one sentence that need to be decomposed.

3 Proposed Method

Our question decomposition module consists of two main process, which are sentence boundary and type recognition and ignored words recognition.

3.1 Data

We used the data from Mahendra et al. [4], which is a collection of patient-doctor question-answer pair taken from five different health consultation forum. For evaluation, we randomly sample and manually annotate 200 question for sentence boundary and sentence type recognition, along with ignored word recognition. More detail specification of annotated data is shown in Tables 1 and 2. To represent the class and position of token in sequence, we use the IOB tag.

Table 1. Specification of annotated sentences

Sentence type	Number of sentence
Background	714
Ignore	359
Question	305
Total	1378

[2] http://www.nlm.nih.gov/research/umls/.

Table 2. Specification of annotated tokens

Tag	Number of token
B-BACKGROUND	714
I-BACKGROUND	8752
B-IGNORE	359
I-IGNORE	928
B-QUESTION	305
I-QUESTION	2668
Total	13726

3.2 Part-of-Speech Tagging

Part-of-speech (POS) tagging is one of data processing steps we apply in our experiment. Given a sequence of words (token), the task is to determine a proper part-of-speech tag for each token. We use sequence labeling approach to do POS tagging, specifically linear chain Conditional Random Field (CRF) model. We use the dataset[3] and tagset definition provided by Dinakaramani et al. [1].

3.3 Sentence Boundary and Type Recognition

Our system take a sentence written by Indonesian patient as an input. We have found that informal language is a common case, such as use of abbreviated word or incorrect/ambiguous sentence boundary. Thus, one of our main task is to determine the boundary of sentences given a long sequence of token. The next task is to classify the type of each sentence. The output of this process is a list of sentence (which is a list of token) with the corresponding type. We define three type of sentence, which are described below.

1. Background: sentence that shows useful contextual information, but doesn't contain question.
2. Question: sentence that contains one or more clause that express a question from user.
3. Ignored: sentence that does not contain any useful information or question and can be ignored.
   Example: "Selamat pagi, dokter" (*Good morning, doctor*)

The method proposed by Mahendra et al. [4] is used as the baseline in our experiment. This baseline method is composed by rules to split sentence and Support Vector Machine model to classify type of each sentence.

We are interested in sequence labeling approach for this task. We propose two-level prediction method which consists of predicting boundary and type separately, and we called this as strategy A. To be more precise, in this strategy,

[3] https://github.com/famrashel/idn-tagged-corpus.

first, we determine which token is the boundary of sentences. Afterwards, the type of each sentence is then determined.

After studying strategy A, we see the potential of a more simple yet potentially more robust end-to-end model. In multiple stage prediction like strategy A, the failure in earlier stage is propagated to the next, which might cause weak overall performance. Thus, we propose strategy B. In this strategy, the prediction of boundary and type of each token is done in one sequence labeling process (Fig. 1).

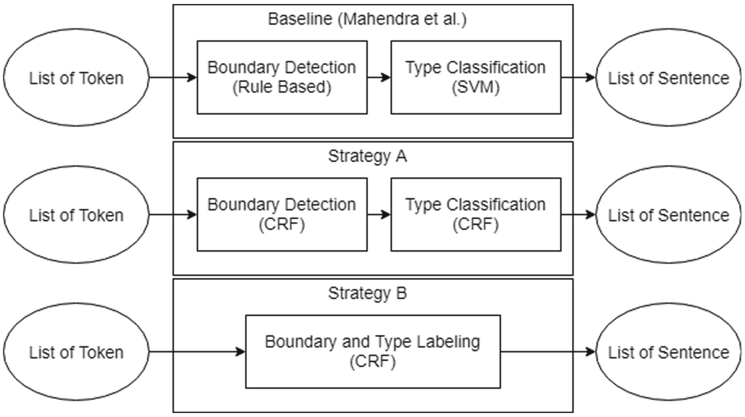

Baseline (Mahendra et al.)
List of Token
Boundary Detection (Rule Based)
Type Classification (SVM)
List of Sentence
Strategy A
List of Token
Boundary Detection (CRF)
Type Classification (CRF)
List of Sentence
Strategy B
List of Token
Boundary and Type Labeling (CRF)
List of Sentence

Fig. 1. Sentence recognition strategies

We use various features for our model, whether on strategy A or B. Below, we list the feature used for sentence boundary detection in strategy A.

1. Features regarding the token itself:
   (a) token itself
   (b) token in lower form
   (c) whether the token is digit
   (d) whether the first character is capital letter
   (e) whether all character is capital letter
2. Feature regarding the position of the token:
   (a) position of the token
   (b) whether the token is the first token in the text
   (c) whether the token is the last token in the text
3. Features regarding neighboring token:
   (a) the token after
   (b) the token before

   (c) whether the token is one of the following: period, comma, exclamation mark, question mark
4. Part-of-Speech
5. Website where the data from
   Every medical forum has their own unique pattern in the data. For example, the question text from Detik Health[4] usually append the biodata of the user which post the consultation question.

For illustration purpose, consider the following example:

- Pagi dok, saya Flu, Apa obatnya?
  (`morning doc, i cought a Flu, What is the medication?`)

The example above consists of three sentences. Note that there are incorrect uses of grammar and it is a very common case in our data. The first sentence ("Pagi dok,") is just a greeting which could be ignored. The second sentence ("saya Flu,") is a background sentence which contain useful symptom information. The last sentence is a question. The IOB tags for each token is shown in Table 3.

Table 3. Example for feature extraction simulation

Sentence 1	Token	Pagi	dok	,
	POS	NN	NN	Z
Sentence 2	Token	saya	**Flu**	,
	POS	PRP	NN	Z
Sentence 3	Token	Apa	obatnya	?
	POS	WH	NN	Z

To give an example, the following will describe all features for boundary detection with strategy A extracted from the token "Flu" (marked in bold) in Table 3.

1. Features regarding the token itself:
   (a) token itself: "Flu"
   (b) token in lower form: "flu"
   (c) whether the token is digit: *false*
   (d) whether the first character is capital letter: *true*
   (e) whether all character is capital letter: *false*
2. Feature regarding the position of the token:
   (a) position of the token: 5/9
   (b) whether the token is the first token in the text: *false*
   (c) whether the token is the last token in the text: *false*

[4] https://health.detik.com/.

3. Features regarding neighboring token:
   (a) the token after: ","
   (b) the token before: "saya"
   (c) whether the token is one of [period, comma, exclamation mark, question mark]: *false*
4. Part-of-Speech: "NN" which stands for noun
5. Website where the data from: "Detik Health"

To classify the type of each sentence in strategy A, the following features are used.

1. Unigram
2. Feature regarding tokens in the sentence:
   (a) quantity of token
   (b) whether the sentence contain question mark
   (c) whether the sentence contain question word, such as "apa" (*what*) and "bagaimana" (*how*)
3. Position of the sentence in the text

To give an example, the following will describe value of each feature for sentence classification with strategy A when extracted from second sentence (marked in bold) in Table 3.

1. Unigram: "saya", "Flu", and ","
2. Feature regarding tokens in the sentence:
   (a) quantity of token: 3
   (b) contain question mark: *false*
   (c) contain question word: *false*
3. Position of the sentence in the text: 2/3

In strategy B, the feature we extract for boundary and type labeling is specified below.

1. Features regarding the token itself:
   (a) token itself
   (b) token in lower form
   (c) whether the token is digit
   (d) whether the first character is capital letter
   (e) whether all character is capital letter
2. Feature regarding the position of the token:
   (a) position of the token
   (b) whether the token is the first token in the text
   (c) whether the token is the last token in the text
3. Features regarding neighboring token:
   (a) the token after
   (b) the token before
   (c) whether the token is one of the following: period, comma, exclamation mark, question mark

4. Part-of-Speech
5. Website where the data from
6. Dictionary of medical terminology
   We use medical terminologies taken from Standar Kompetensi Dokter Indonesia 2012[5] (Indonesian Standard Competency of Doctor 2012).
7. Abbreviation dictionary
   We use dictionary of abbreviation of Indonesian words from [9].

To give an example, the following will describe the value of the feature for strategy B when extracted from the token "Flu" (marked in bold) in Table 3.

1. Features regarding the token itself:
   (a) token itself: "Flu"
   (b) token in lower form: "flu"
   (c) is digit: *false*
   (d) first character is capital letter: *true*
   (e) all character is capital letter: *false*
2. Feature regarding the position of the token:
   (a) position of the token: 5/9
   (b) is the first token in the text: *false*
   (c) is the last token in the text: *false*
3. Features regarding neighboring token:
   (a) the token after: ","
   (b) the token before: "saya"
   (c) the token is one of [period, comma, exclamation mark, question mark]: *false*
4. Part-of-Speech: "NN" (noun)
5. Website where the data from: Detik Health
6. Dictionary of medical terminology: *true* (is a medical terminology)
7. Abbreviation dictionary: not abbreviated

3.4 Ignored Word Recognition

Our user often use word that doesn't contain any useful contextual information. One of the common case is greeting words, such as "dok" (a form of greeting to doctor). Given background or question sentence, the task is to identify words that could be ignored.

We propose sequence labeling approach for this task. We use linear chain CRF as our learning model. The feature we extract for our model is listed below.

1. Features regarding the token itself:
   (a) token itself
   (b) token in lower form
   (c) whether the token begins with capital letter
2. Features regarding neighboring token:
   (a) the token after
   (b) the token before
3. The position of token in text

[5] http://pd.fk.ub.ac.id/wp-content/uploads/2014/12/SKDI-disahkan.pdf.

4 Result and Analysis

4.1 Sentence Boundary and Type Recognition

The performance of strategy A and B is shown in Table 4. This result is obtained by training all strategy end-to-end, which means when the strategy composed of two steps like strategy A, the second step is evaluated using the result of the first step. In addition, we also compare it with baseline method [4], which applied using the same dataset we use in our experiment. We evaluate each of our proposed strategies using 10-fold cross-validation.

Table 4. Performance of Strategy A and B in end-to-end evaluation. For comparison, we also present the baseline method performance. Tag: BG = Background, IG = Ignore, QS = Question

Tag	Precision			Recall			F1		
	Base	A	B	Base	A	B	Base	A	B
B-BG	0.42	0.72	0.83	0.23	0.65	0.71	0.30	0.68	0.77
I-BG	0.67	0.90	0.92	0.36	0.91	0.97	0.46	0.91	0.94
B-IG	0.25	0.76	0.93	0.47	0.65	0.78	0.33	0.70	0.85
I-IG	0.07	0.72	0.88	0.61	0.60	0.76	0.13	0.66	0.82
B-QS	0.00	0.71	0.83	0.00	0.64	0.77	0.00	0.67	0.80
I-QS	0.00	0.76	0.89	0.00	0.82	0.85	0.00	0.79	0.87
Average	0.46	0.84	0.91	0.29	0.85	0.91	0.33	0.84	0.91

We first discuss the performance of strategy A in more detail. Strategy A can be evaluated per step (sentence boundary detection and sentence type classification) independently. This means when we evaluate the sentence type classification, we assume that the boundary is correct. The result of each steps in strategy A is shown in Table 5 and Table 6. We also compare strategy A with baseline method in the same evaluation style.

Table 5. Performance of Sentence Boundary Detection on Strategy A in independent evaluation

Tag	Precision		Recall		F1	
	Base	A	Base	A	Base	A
BEGIN	0.85	0.88	0.67	0.78	0.75	0.83
NOT_BEGIN	0.96	0.98	0.99	0.99	0.98	0.98
Average / Total	0.95	0.97	0.96	0.97	0.95	0.97

Table 6. Performance of Sentence Type Classification on Strategy A in independent evaluation

Sentence type	Precision		Recall		F1	
	Base	A	Base	A	Base	A
BACKGROUND	0.84	0.91	0.93	0.97	0.88	0.94
IGNORE	0.82	0.95	0.77	0.89	0.80	0.92
QUESTION	0.95	0.95	0.79	0.88	0.86	0.91
Average/Total	0.86	0.93	0.86	0.93	0.86	0.93

In strategy B, we found that 71.62% of the sentences are exact match, specifically 232 out of 305 (76.06%) question sentence, 482 out of 714 (76.04%) background sentence, and 273 out of 359 (67.50%) ignored sentence.

There are also cases where the sentence are predicted partially. For example, there are predicted sentences that are cutoff in the beginning and/or end. We found 70 out of total 1378 (28.37%) sentences are partially predicted, in which 12 of them are question sentences, 47 are background sentences, and 11 are ignored sentences. The composition of those 70 partial cases is the following: 30 cases cutoff in the beginning, 36 cases cutoff in the end, and 4 cases cutoff in both beginning and end. Furthermore, the average number of token that are cutoff is 6.5 tokens in the beginning of the sentence, 21.5 in the end, and 21.5 in both the beginning and end of the sentence. Overall, the average number of cutoff token is 8.7 tokens.

Other than exact match and partial match, we also observe cases where the correctly predicted sentence extend into the neighboring sentence, whether at the beginning or at the end of the sentence. In other words, the predicted sentence contain parts of other sentence that it shouldn't. Out of 1378 sentences, we found 341 cases like this. It composes of 138 cases where the sentence extends at the beginning into prior sentence, 126 cases where the sentence extends at the end into next sentence, and 70 where the sentence extends at the beginning and end. In average, it extends 9.2 tokens at the beginning and 16.7 tokens at the end of the sentence. More than 90% of the excessive tokens belong to background sentence, which means that background sentence are more likely to extend into neighboring sentence than other types of sentence.

We also observe the cases where a sentence is not predicted, not even partially. We count 104 sentences that fall into this case.

Other than cases mentioned before, we also found cases where the sentence boundary is predicted correctly, but the category is predicted incorrectly. We count that 76 sentences fall into this category out of 1063 sentences that the boundary is predicted correctly (7.14%).

4.2 Ignored Word Recognition

We evaluate the proposed ignored word recognition with 10-fold cross-validation. The result is shown in Table 7. In our observation of the results, most of the incorrect prediction is caused by inabilities of our proposed model to recognize the difference between word as a subject and word as a greeting. To give more illustration, consider the following examples:

– **dokter** urologi menerangkan bahwa ginjal anak saya membengkak.
  (`urology doctor explain that my child's kidney swell.`)
– apa obat yang cocok, **dokter**?
  (`what is the suitable medicine, doctor?`)

The word "doctor" in the first example is a subject, but the word "doctor" in the second example is a greeting word. We define greeting word, such as "doctor" in the second example, as word that can be ignored.

Table 7. Performance of ignored word recognition

Precision	Recall	F1-Score
0.94	0.86	0.90

5 Conclusion

We have proposed the use of sequence labeling approach to find question and contextual medical information from text, along with part of the text that can be ignored. In general, our module receives complex health question as an input and produce list of background, question, and ignored sentence as an output. We use linear-chain Conditional Random Field as our model and analyze several strategy in our proposed method. The evaluation shows that our sequence labeling model achieves F1 score of 0.83 in detecting boundary of sentence and F1 score of 0.93 when classifying type of sentence. Furthermore, when detecting word that could be ignored in sentence, the proposed method achieves F1 score of 0.90.

For future work, we may leverage the keyphrase extraction model [7] to improve the current system to detect the question. Furthermore, we are interested in designing end-to-end model using deep learning architecture, e.g. Long Short-Term Memory (LSTM) [10] or BERT Transformer-based model [11,12], to solve this task. We are also curious to see how the model perform in a much larger size of dataset.

References

1. Dinakaramani, A., Rashel, F., Luthfi, A., Manurung, R.: Designing an Indonesian part of speech tagset and manually tagged indonesian corpus. In: 2014 International Conference on Asian Language Processing (IALP), pp. 66–69, October 2014. https://doi.org/10.1109/IALP.2014.6973519
2. Evang, K., Basile, V., Chrupała, G., Bos, J.: Elephant: sequence labeling for word and sentence segmentation. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pp. 1422–1426, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. http://www.aclweb.org/anthology/D13-1146
3. Hakim, A.N., Mahendra, R., Adriani, M., Ekakristi, A.S.: Corpus development for Indonesian consumer-health question answering system. In: 2017 International Conference on Advanced Computer Science and Information Systems (ICACSIS) (2017)
4. Mahendra, R., Hakim, A.N., Adriani, M.: Towards question identification from online healthcare consultation forum post in bahasa. In: 2017 International Conference on Asian Language Processing (IALP) (2017)
5. Roberts, K., Masterton, K., Fiszman, M., Kilicoglu, H., Demner-fushman, D.: Annotating question decomposition on complex medical questions. In: Proceedings of LREC (2014)
6. Roberts, K., Fiszman, M., Kilicoglu, H., Demmer-Fushman, D.: Decomposing consumer health questions. In: Proceedings of the 2014 Workshop on Biomedical Natural Languange Processing, pp. 29–37. U.S. National Library of Medicine (2017)
7. Saputra, I.F., Mahendra, R., Wicaksono, A.F.: Keyphrases extraction from user-generated contents in healthcare domain using long short-term memory networks. In: Proceedings of the BioNLP 2018 workshop, pp. 28–34, Melbourne, Australia, July 2018. Association for Computational Linguistics. https://doi.org/10.18653/v1/W18-2304. https://aclanthology.org/W18-2304
8. Sondhi, P., Gupta, M., Zhai, C., Hockenmaier, J.: Shallow information extraction from medical forum data. In Proceedings of the 23rd International Conference on Computational Linguistics: Posters, COLING 2010, pp. 1158–1166. Association for Computational Linguistics, Stroudsburg (2010). http://dl.acm.org/citation.cfm?id=1944566.1944699
9. Wicaksono, A.F., Vania, C., Distiawan, B., Adriani, M.: Automatically building a corpus for sentiment analysis on Indonesian tweets. In Proceedings of the 28th Pacific Asia Conference on Language, Information and Computation, PACLIC 28, Cape Panwa Hotel, Phuket, Thailand, December 12–14 (2014), pp. 185–194 (2014). http://aclweb.org/anthology/Y/Y14/Y14-1024.pdf
10. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997). MIT Press
11. Devlin, J., Chang, M.-W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, vol. 1 (Long and Short Papers), pp. 4171–4186. Association for Computational Linguistics, Minneapolis (2019)
12. Wilie, B., et al.: IndoNLU: benchmark and resources for evaluating Indonesian natural language understanding. In: Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing, pp. 843–857. Association for Computational Linguistics, Suzhou (2020)

Argumentation in Text: Discourse Structure Matters

Boris Galitsky[1], Dmitry Ilvovsky[2](✉), and Dina Pisarevskaya[3]

[1] Oracle Inc., Redwood Shores, CA, USA
boris.galitsky@oracle.com
[2] National Research University Higher School of Economics, Moscow, Russia
dilvovsky@hse.ru
[3] Institute for Systems Analysis, Federal Research Center "Computer Science and Control" of Russian Academy of Sciences, Moscow, Russia

Abstract. We address the problem of argument detection by investigating discourse and communicative text structure. A formal graph-based structure called communicative discourse tree (CDT) is used. It consists of a discourse tree (DT) with additional labels on edges, which stand for verbs. These verbs represent communicative actions. Discourse trees are based on rhetoric relations, extracted from a text according to Rhetoric Structure Theory. The problem is tackled as a binary classification task, where the positive class corresponds to texts with arguments and the negative class corresponds to texts with no argumentation. The feature engineering for the classification task is conducted, deciding which discourse and communicative features are better associated with argumentation. New Intense Argumentation dataset is built and described. Mixed dataset including different types of argumentation and different text genres is collected. Evaluation on this mixed dataset is provided.

Keywords: Argumentation · Discourse · Dataset

1 Introduction

When an author attempts to provide an argument for something, a number of argumentation patterns can be employed. An argument is the key point of any persuasive essay or speech. The target of this paper is to recognize discourse features of text where an author not just shares her point of view but also provides a reason for it and attempts to prove it as well. To systematically extract argumentation patterns, we compile the Intense Argumentation Dataset where authors attempt to back up their complaints with sound argumentation.

Naturally, the text units considered in discourse analysis correspond to argument components, and discourse relations are closely related to argumentative relations. However, the traditional training dataset for rhetoric parsing consists of newspaper articles which do not necessarily involve heavy argumentation, and only relations between adjacent text units are identified. It is still an open question how the proposed discourse relations relate to argumentative relations [2].

To represent the linguistic features of text, we use the following sources: 1) *Rhetoric relations* between the parts of the sentences, obtained as a *discourse tree* [21]; 2) *Speech acts, communicative actions*, obtained as verbs from the VerbNet resource (the verb signatures with instantiated semantic roles). These are attached to rhetoric relations.

The final goal of this ongoing research is to estimate the contribution of each feature type to the problem of argument identification in text fragments.

The main contribution of our work at the current step is the following:

1. We apply the notion of Communicative Discourse Tree (CDT) for the specific text classification task
2. We develop a part of a text classification framework that includes automatic CDT extraction from text paragraphs, tree kernel learning on CDT, kNN learning on CDT based on computing similarity between CDTs.
3. We apply our framework for the binary classification task on the dataset consisting of mixed texts with different types of argumentation and compare the performance of a few learning methods in combination with different features.
4. We built and published new language resourse - Intense Argumentation Dataset containing different patterns of valid and invalid argumentation.

2 Related Work

Most previous work in automated discourse analysis is based on the extracting patterns from the corpora annotated with discourse relations, most notably the Penn Discourse Treebank (PDTB) [32] and the Rhetorical Structure Theory (RST) Discourse Treebank [3]. An extensive corpus of studies has been devoted to RST parsers, but the research on how to leverage RST parsing results for practical NLP problems is rather limited.

It is well known that argumentation and discourse structure of text are strongly related with each other. In [1] authors claim that performing an RST analysis essentially subsumes the task of determining argumentation structure. As it was shown [29] recently RST analysis can in principle support an argumentation analysis. Also an annotation of a discourse structure [41] is a field that is closely related to the annotation of argumentation structures [15].

There are different approaches to argument mining. The basic argument model can be represented with a scheme - a set of statements which contains three elements: a conclusion, a set of premises, and an inference from the premises to the conclusion [38]. Other models were offered in [37] and [6]. In general, all these models refer to an argument as a conclusion (or a claim) and a set of premises (or reasons). Text fragments can be classified into argumentation schemes - templates for typical arguments. So argument mining can consist of the following steps: identifying argumentative segments in text [19,20,36], clustering and classifying arguments [24], determining argument structure [10,17], getting predefined argument schemas [4]. Recent works in argumentation mining study different features related to discourse, considering arguments which support claims [9,11,30], the relationship between argumentation structure and discourse structure (in terms of Rhetorical Structure Theory) is also the focus of contemporary research [31].

Previously, annotation schemes and approaches for identifying arguments in different domains have been developed, including [27] for legal documents, [40] for newspapers and court cases, [5] for policy modelling, and [33] for persuasive essays.

The concept of automatically identifying argumentation schemes was first discussed in [39] and [4]. Most of the approaches focus on the identification and classification of argument components. In [10] authors investigate argumentation discourse structure of the specific type of communication - online interaction threads. In [16] three types of argument structure identification are combined: linguistic features, topic changes and machine learning.

3 Communicative Discourse Tree

3.1 Case Study

We consider a controversial article published in Wall Street Journal about Theranos[1], a company providing healthcare services, and the company rebuttal.

RST represents texts by labeled hierarchical structures, called Discourse Trees (DTs). The leaves of a DT correspond to contiguous atomic text spans, Elementary Discourse Units (EDUs). EDUs are clause-like units that serve as building blocks. RST relations connect adjacent EDUs to form next-level discourse units represented by internal nodes. These nodes are in turn subjects to linking by RST relations. Discourse units linked by a rhetorical relation are further distinguished based on their relative importance in the text: nuclei are the core parts of the relation and satellites are peripheral or supportive ones.

We build an RST representation of the arguments and observe if a DT is capable of indicating whether a paragraph communicates both a claim and an argumentation that backs it up. We will then explore what needs to be added to a DT so that it is possible to judge if it expresses an argumentation pattern or not.

DTs and their images in this case study are obtained by the software of [35]. This is what happened according to Carreyrou[2]:

```
"Since October [2015], the Wall Street Journal has published
a series of anonymously sourced accusations that inaccurately
portray Theranos. Now, in its latest story (‘‘U.S. Probes
Theranos Complaints,’’ Dec. 20), the Journal once again is
relying on anonymous sources, this time reporting two
undisclosed and unconfirmed complaints that allegedly were
filed with the Centers for Medicare and Medicaid Services
(CMS) and U.S. Food and Drug Administration (FDA)."
```

[1] https://theranos.com/news/posts/wall-street-journal-letter-to-the-editor.
[2] http://www.wsj.com/articles/theranos-has-struggled-with-blood-tests.

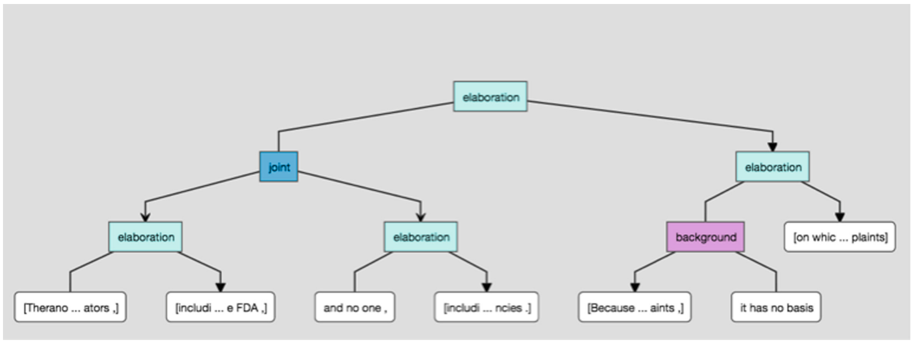


Fig. 1. Discourse tree with multiple rhetoric relations

We as a readers understand that Theranos attempts to rebuke the claim of WSJ. But Fig. 1 demonstrates that just from a DT and multiple rhetoric relations of *elaboration* and a single instance of *background*, it is unclear whether an author argues with his opponents or enumerating some observations.

```
"Theranos remains actively engaged with its regulators,
including CMS and the FDA, and no one, including the Wall
Street Journal, has provided Theranos a copy of the alleged
complaints to those agencies. Because Theranos has not seen
these alleged complaints, it has no basis on which to
evaluate the purported complaints."
```

For the following paragraph Fig. 2 shows the DT with additional communicative actions labels which help to identify presence of argumentation. When arbitrary communicative actions are attached to DT as labels of its terminal arcs, it becomes clear that the author is trying to bring her point across and not merely sharing a fact.

```
"But Theranos has struggled behind the scenes to turn the
excitement over its technology into reality. At the end of
2014, the lab instrument developed as the linchpin of its
strategy handled just a small fraction of the tests then sold
to consumers, according to four former employees."
```

3.2 Definition

As it can be seen from this example to show the structure of arguments we need to know the discourse structure of interactions between agents, and what kind of interactions they are. We do not need to know domain of interaction (here, health), the subjects of these interaction (the company, the journal, the agencies), what are the entities, but we need to take into account mental, domain-independent relations between them.

Communicative discourse tree (CDT) [7] *is a DT with labels for arcs which are the VerbNet expressions for verbs which are communicative actions*

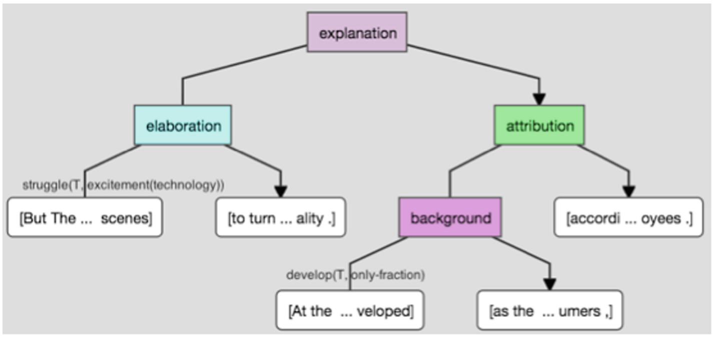


Fig. 2. Discourse tree with attached communicative actions

(CA). The arguments of verbs are substituted from text according to VerbNet frames. Arguments of verbs are substituted from text according to VerbNet frames. The first, possibly second and third argument are instantiated by agents and the last ones by noun or verb phrases. These phrases are subjects of communicative action.

CA can take the form of verb (agent, subject, cause) where verb characterizes, for example, some sort of interaction between a customer and company in a complaint scenario (e.g., *explain, confirm, remind, disagree, deny*), agent identifies either the customer or the company, subject refers to the information transmitted or object described, and cause refers to the motivation or explanation for the subject. A communicative action associated with some customer claim such as *I disagreed with the overdraft fee you charged me because I made a bank deposit well* in advance would be represented as: *disagree (customer, overdraft fee, I made a bank deposit well in advance).* VerbNet frames are used to apply the computational part of Speech Act theory to discourse analysis, formalizing CAs.

For the details of DTs we refer the reader to [12], and for VerbNet Frames to [14]. To build CDT **automatically** we combined together discourse parsers [13,35] with our own modules focused on extracting and information from VerbNet [28] into one Java-oriented system. Our project and examples of CDT representation can be found at GitHub.

4 Text Classification Settings

To evaluate the contribution of our sources, we use two types of learning on CDT graph representations of a paragraph. 1) Nearest Neighbour (kNN) learning with explicit engineering of graph descriptions. We measure similarity as an overlap between the graph representation of a given text and that of a given element of a training set. 2) Statistical tree kernel learning of structures with implicit feature engineering.

We consider standalone discourse trees and scenario graphs built on communicative actions extracted from the text as well as full CDT graphs.

Our family of pre-baseline approaches is based on keywords and keywords statistics. Since mostly lexical and length-based features are reliable for finding poorly supported arguments [34], we combined non-name entities as features together with the number of tokens in the phrase which potentially expresses argumentation.

4.1 Nearest Neighbour

To predict the label of the text, once the CDT is built, one needs to compute its similarity with CDTs for the positive class and verify that it is lower than similarity to the set of CDTs for its negative class. Similarity between CDT's is defined by means of **maximal common sub-CDTs** [7]. Formal definitions of labeled graphs and domination relation on them used for construction of this operation can be found, e.g., in [8]. To handle meaning of words expressing the subjects of edge label, we also apply word2vec models [22,23]. Similarity of meaning is calculated on a word-by-word basis: if two words are in the same syntactic role, only then they are matched. For computing maximal common sub-CDT we developed our own programming module which is also integrated into the project mentioned above.

4.2 SVM Tree Kernel

In this study we extend the tree kernel definition for the CDT, augmenting DT kernel by the information on communicative actions [7]. A CDT can be represented by a vector of integer counts of each sub-tree type (without taking into account its ancestors). The terms for Communicative Actions as labels are converted into trees which are added to respective nodes for RST relations. For Elementary Discourse Units (EDUs) as labels for terminal nodes only the phrase structure is retained: we label the terminal nodes with the sequence of phrase types instead of parse tree fragments. For the evaluation purpose we used Tree Kernel builder tool [25].

5 Datasets

5.1 New Intense Argumentation Dataset

The set of tagged customer complaints about financial services is available at GitHub.

The purpose of this dataset is to collect texts where authors do their best to bring their points across by employing all means to show that they are right and their opponents are wrong. Complainants are emotionally charged writers who describe problems they encountered with a financial service and how they attempted to solve it. Raw complaints are collected from PlanetFeedback.com for a number of banks submitted in 2006–2010. Four hundred complaints are manually tagged with respect to perceived complaint validity, proper argumentation and detectable misrepresentation.

Judging by complaints, most complainants are in genuine distress due to a strong deviation between what they expected from a service, what they received and how it was communicated. Most complaint authors report incompetence, flawed policies, ignorance, indifference to customer needs and misrepresentation from the customer service personnel. The authors are frequently exhausted from communicative means available to them; they could be confused, seeking recommendation from other users. The focus of a complaint is a proof that the proponent is right and her opponent is wrong, resolution proposal and a desired outcome.

Multiple argumentation patterns are used in complaints. The most frequent is a deviation from what has happened from what was expected, according to common sense. This pattern covers both valid and invalid argumentation. The second in popularity argumentation patterns cites the difference between what has been promised (advertised, communicated) and what has been received or actually occurred. This pattern also mentions that the opponent does not play by the rules (valid pattern).

A high number of complaints are explicitly saying that bank representatives are lying. Lying includes inconsistencies between the information provided by different bank agents, factual misrepresentation and careless promises (valid pattern).

Another reason complaints arise is due to rudeness of bank agents and customer service personnel. Customers cite rudeness in both cases, when the opponent point is valid or not (and complaint and argumentation validity is tagged accordingly). Even if there is neither financial loss or inconvenience the complainants disagree with everything a given bank does, if they been served rudely (invalid pattern).

Complainants cite their needs as reasons bank should behave in certain ways. A popular argument is that since the government via taxpayers bailed out the banks, they should now favor the customers (invalid pattern).

We refer to this dataset as Intense because of the amount, strength and emotional load of customer complaints. For a given topic such as insufficient funds fee, this dataset provides many distinct ways of argumentation that this fee is unfair. Therefore, Intense Argumentation dataset allows for systematic exploration of the topic-independent clusters of argumentation patterns and observe a link between argumentation type and overall complaint validity. Other argumentation datasets including legal arguments, student essays[3], internet argument corpus[4], fact-feeling dataset[5], political debates have a strong variation of topics so that it is harder to track a spectrum of possible argumentation patterns per topic. Unlike professional writing in legal and political domains, authentic writing of complaining users have a simple motivational structure, a transparency of their purpose and occurs in a fixed domain and context. In the Intense Argumentation Dataset, the arguments play a critical rule for the well-being of the authors, subject to an unfair charge of a large amount of money or eviction from

[3] https://www.ukp.tu-darmstadt.de/fileadmin/user_upload/Group.../EACL2017-Stab.pdf.

[4] https://nlds.soe.ucsc.edu/iac2.

[5] https://nlds.soe.ucsc.edu/factfeel.

home. Therefore, the authors attempt to provide as strong argumentation as possible to back up their claims and strengthen their case.

Using our Intense Dataset, one can find correlation between argumentation validity, truthfulness and overall complaint validity. If a complaint is not truthful it is usually invalid: either a customer complains out of a bad mood or she wants to get compensation. However, if the complaint is truthful it can easily be invalid, especially when arguments are flawed. When an untruthful complaint has valid argumentation patterns, it is hard for an annotator to properly assign it as valid or invalid. Three annotators worked with this dataset, and inter-annotator agreement exceeds 80%.

5.2 Additional Evaluation Dataset

For the particular task described in this paper we collected a large dataset, which includes the Intense Argumentation Dataset described in the previous section.

Evaluation dataset was divided into two parts: "positive" and "negative". Texts from the first part are expected to contain any kind of argumentation inside. We formed the positive dataset from a few sources to make it non-uniform and pick together different styles, genres and argumentation types. First we used a portion of data where argumentation is frequent, e.g. opinionated data from newspapers such as The New York Times (1400 articles), The Boston Globe (1150 articles), Los Angeles Times (2140) and others (1200).

As it was mentioned earlier we also used our new Intense Dataset. Besides, we use the text style & genre recognition dataset [18] which has a specific dimension associated with argumentation (the section [ted] "Emotional speech on a political topic with an attempt to sound convincing"). And we finally add some texts from standard argument mining datasets where presence of arguments is established by annotators: "Fact and Feeling" dataset [26] (680 articles) and dataset "Argument annotated essays v.2" [33](430 articles).

Negative part of the dataset consists of texts written in a neutral manner. We use Wikipedia (3500 articles), factual news sources (Reuters feed with 3400 articles) and also [18] dataset including such sections of the corpus "Instructions for how to use software" (320 articles); [tele], "Instructions for how to use hardware" (175 articles); [news], "A presentation of a news article in an objective, independent manner" (220 articles), and other mixed datasets without argumentation (735 articles). Both datasets include 8800 texts.

We used Amazon Mechanical Turk to confirm that the positive dataset includes argumentation in a commonsense view, according to the employed workers. Twelve workers who had the previous acceptance score of above 85% were assigned the task to label.

6 Evaluation

For the evaluation we split out dataset into the training and test part in proportion of 4:1 and balanced the split with respect to the label and to the source.

Table 1. Evaluation results: Nearest Neighbour classification

Method/sources	Precision	Recall	F1
Approach based on keywords	57.2	53.1	55.07
Naive Bayes	59.4	55.0	57.12
Graph-based kNN for reduced CDT (DT only)	65.6	60.4	62.89
Graph-based kNN for reduced CDT (CA only)	62.3	59.5	60.87
Graph-based kNN for full CDT	**83.1**	**75.8**	**79.28**

Table 2. Evaluation results: SVM TK

Method/sources	Precision	Recall	F1
SVM TK for reduced CDT (DT only)	63.6	62.8	63.20
SVM TK for full CDT	82.4	79.61	79.61

Extremely naive approach is just relying on keywords (bag-of-words) to figure out a presence of argumentation. The hypothesis here is that people use different words to describe facts vs words to back them up and explicitly provide argumentation. Usually, a couple of communicative actions so that at least one has a negative sentiment polarity (related to an opponent) are sufficient that argumentation is present. In Table 1, we see that this naive approach is outperformed by the top performing CDT approach by 22%. A Naive Bayes classifier delivers just 2% improvement. One can observe that for nearest neighbor learning DT and scenario graphs based on CA indeed complement each other, delivering f-measure of full CDT 17% above the former and 19% above the latter. Just CA delivered worse results than the standalone DT.

Nearest neighbor learning for full CDT achieves slightly lower performance than SVM TK for full CDT, but the former gives interesting examples of sub-trees which are typical for argumentation, and the ones which are shared among the factual data. The number of the former groups of CDT sub-trees is naturally significantly higher (Table 2).

Table 3. Classification results for individual sources of argumentation (F1 measure)

Method/sources	Newspapers	Textual complaints (intense dataset)	Style and genre recognition	Fact and feeling
Approach based on keywords	52.3	55.2	53.7	54.8
Naive Bayes	57.1	58.3	57.2	59.4
Reduced CDT (DT only)	66.0	63.6	67.9	66.3
Reduced (CA only)	64.5	60.3	62.5	60.9
Full CDT (DT + CA)	77.1	78.8	80.3	79.2

Table 3 shows the SVM TK argument detection results per source. As a positive set, we now take individual source only. The negative set is formed from the same sources but reduced in size to match the size of a smaller positive set. The cross-validation settings are analogous to our assessment of the whole positive set. We did not find correlation between the peculiarities of a particular domain and contribution of discourse-level information to argument detection accuracy. At the same time, all these four domains show monotonic improvement when we proceed from Keywords and Naive Bayes to CDT. Since all four sources demonstrate the improvement of argument detection rate due to CDT, we conclude that the same is likely for other source of argumentation-related information.

7 Conclusion

In this study we addressed an issue of argumentation detection in text using it communicative and discourse structure. We described a few representations that capture this kind of structure. We then compared two learning methods working over these representations. Performances of these learning methods showed that the bottleneck of text classification based on textual discourse information is in the representation means, not in the learning method itself. Comparing inductive learning results with the kernel-based statistical learning, relying on the same information allowed us to perform more concise feature engineering than either approach would do. We see that text classification, based on Nearest Neighbour learning, shows better results with Communicative discourse tree features than with only Discourse Tree features or with only communicative actions features. We also built and published a set of tagged customer complaints about financial services which can be used for the future case studies and research in argumentation mining and text classification.

Acknowledgments. This article was prepared within the framework of the Basic Research Program at the National Research University Higher School of Economics (HSE) and supported within the framework of a subsidy by the Russian Academic Excellence Project ‘5-100’. It was supported by the RFBR grants 16-29-12982, 16-01-00583.

References

1. Azar, M.: Argumentative text as rhetorical structure: an application of rhetorical structure theory. Argumentation **13**(1), 97–144 (1999)
2. Biran, O., Rambow, O.: Identifying justifications in written dialogs by classifying text as argumentative. Int. J. Semant. Comput. **5**(04), 363–381 (2011)
3. Carlson, L., Marcu, D., Okurowski, M.E.: Building a discourse-tagged corpus in the framework of rhetorical structure theory. In: van Kuppevelt, J., Smith, R.W. (eds.) Current and New Directions in Discourse and Dialogue. Text, Speech and Language Technology, vol. 22, pp. 85–112. Springer, Dordrecht (2003). https://doi.org/10.1007/978-94-010-0019-2_5

4. Feng, V.W., Hirst, G.: Classifying arguments by scheme. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1, pp. 987–996. Association for Computational Linguistics (2011)
5. Florou, E., Konstantopoulos, S., Koukourikos, A., Karampiperis, P.: Argument extraction for supporting public policy formulation. In: Proceedings of the 7th Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities, pp. 49–54. Citeseer (2013)
6. Freeman, J.B.: Dialectics and the Macrostructure of Arguments: A Theory of Argument Structure, vol. 10. Walter de Gruyter, Berlin (1991)
7. Galitsky, B.: Discovering rhetorical agreement between a request and response. Dialogue Discourse, 167–205 (2017)
8. Ganter, B., Kuznetsov, S.O.: Pattern structures and their projections. In: Delugach, H.S., Stumme, G. (eds.) ICCS-ConceptStruct 2001. LNCS (LNAI), vol. 2120, pp. 129–142. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44583-8_10
9. Ghosh, D., Khanam, A., Han, Y., Muresan, S.: Coarse-grained argumentation features for scoring persuasive essays. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pp. 549–554. Association for Computational Linguistics (2016)
10. Ghosh, D., Muresan, S., Wacholder, N., Aakhus, M., Mitsui, M.: Analyzing argumentative discourse units in online interactions. In: Proceedings of the First Workshop on Argumentation Mining, pp. 39–48 (2014)
11. Habernal, I., Gurevych, I.: Which argument is more convincing? Analyzing and predicting convincingness of web arguments using bidirectional LSTM. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 1589–1599. Association for Computational Linguistics (2016)
12. Joty, S., Carenini, G., Ng, R.T.: A novel discriminative framework for sentence-level discourse analysis. In: Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pp. 904–915. Association for Computational Linguistics (2012)
13. Joty, S., Moschitti, A.: Discriminative reranking of discourse parses using tree kernels. a) A 4(5), 6 (2014)
14. Kipper, K., Korhonen, A., Ryant, N., Palmer, M.: A large-scale classification of English verbs. Lang. Resour. Eval. **42**(1), 21–40 (2008)
15. Kirschner, C., Eckle-Kohler, J., Gurevych, I.: Linking the thoughts: analysis of argumentation structures in scientific publications. In: Proceedings of the 2nd Workshop on Argumentation Mining, pp. 1–11 (2015)
16. Lawrence, J., Reed, C.: Combining argument mining techniques. In: NAACL HLT 2015, p. 127 (2015)
17. Lawrence, J., Reed, C., Allen, C., McAlister, S., Ravenscroft, A.: Mining arguments from 19th century philosophical texts using topic based modelling. In: Proceedings of the First Workshop on Argumentation Mining, pp. 79–87. Association for Computational Linguistics (2014)
18. Lee, D.Y.: Genres, registers, text types, domains and styles: clarifying the concepts and navigating a path through the BNC jungle. Technology **5**, 37–72 (2001)
19. Levy, R., Bilu, Y., Hershcovich, D., Aharoni, E., Slonim, N.: Context dependent claim detection. In: Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pp. 1489–1500. Dublin City University and Association for Computational Linguistics (2014)

20. Lippi, M., Torroni, P.: Context-independent claim detection for argument mining. In: Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, pp. 185–191 (2015)
21. Mann, W.C., Thompson, S.A.: Rhetorical structure theory: toward a functional theory of text organization. Text-Interdisc. J. Study Discourse **8**(3), 243–281 (1988)
22. Mikolov, T., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems (2013)
23. Mikolov, T., Chen, K., Corrado, G.S., Dean, J.A.: Computing numeric representations of words in a high-dimensional space (19 May 2015), uS Patent 9,037,464
24. Misra, A., Anand, P., Fox Tree, J.E., Walker, M.: Using summarization to discover argument facets in online idealogical dialog. In: Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 430–440. Association for Computational Linguistics (2015)
25. Moschitti, A.: Efficient convolution kernels for dependency and constituent syntactic trees. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) ECML 2006. LNCS (LNAI), vol. 4212, pp. 318–329. Springer, Heidelberg (2006). https://doi.org/10.1007/11871842_32
26. Oraby, S., Reed, L., Compton, R., Riloff, E., Walker, M., Whittaker, S.: And that's a fact: Distinguishing factual and emotional argumentation in online dialogue. In: NAACL HLT 2015, p. 116 (2015)
27. Palau, R.M., Moens, M.F.: Argumentation mining: the detection, classification and structure of arguments in text. In: Proceedings of the 12th International Conference on Artificial Intelligence and Law, pp. 98–107. ACM (2009)
28. Palmer, M.: Semlink: linking propbank, verbnet and framenet. In: Proceedings of the Generative Lexicon Conference, pp. 9–15 (2009)
29. Peldszus, A., Stede, M.: Rhetorical structure and argumentation structure in monologue text. In: Proceedings of the 3rd Workshop on Argument Mining, ACL 2016, pp. 103–112 (2016)
30. Peldszus, A., Stede, M.: An annotated corpus of argumentative microtexts. In: Argumentation and Reasoned Action: Proceedings of the 1st European Conference on Argumentation, Lisbon 2015, vol. 2, pp. 801–815. College Publications, London (2016a)
31. Peldszus, A., Stede, M.: Rhetorical structure and argumentation structure in monologue text. In: Proceedings of the 3rd Workshop on Argumentation Mining, pp. 103–112. Association for Computational Linguistics (2016b)
32. Prasad, R., et al.: The Penn discourse treebank 2.0. In: LREC. Citeseer (2008)
33. Stab, C., Gurevych, I.: Recognizing the absence of opposing arguments in persuasive essays. In: ACL 2016, p. 113 (2016)
34. Stab, C., Gurevych, I.: Recognizing insufficiently supported arguments in argumentative essays. In: Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2017), pp. 980–990. Association for Computational Linguistics, April 2017
35. Surdeanu, M., Hicks, T., Valenzuela-Escárcega, M.A.: Two practical rhetorical structure theory parsers. In: Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations, pp. 1–5 (2015)
36. Swanson, R., Ecker, B., Walker, M.: Argument mining: extracting arguments from online dialogue. In: Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue, pp. 217–226. Association for Computational Linguistics, Prague, Czech Republic (2015)

37. Toulmin, S.E.: The Uses of Argument. Cambridge University Press, Cambridge (1958)
38. Walton, D.: Argumentation theory: a very short introduction. In: Simari, G., Rahwan, I. (eds.) Argumentation in Artificial Intelligence, pp. 1–22. Springer, Boston (2009). https://doi.org/10.1007/978-0-387-98197-0_1
39. Walton, D., Reed, C., Macagno, F.: Argumentation Schemes. Cambridge University Press, Cambridge (2008)
40. Walton, D.N.: Argumentation Schemes for Presumptive Reasoning. L. Erlbaum Associates, Mahwah (1996)
41. Webber, B., Egg, M., Kordoni, V.: Discourse structure and language technology. Nat. Lang. Eng. **18**(04), 437–490 (2012)

A Deep Learning Approach for Multimodal Deception Detection

Gangeshwar Krishnamurthy[1], Navonil Majumder[2], Soujanya Poria[2], and Erik Cambria[3](✉)

[1] Artificial Intelligence Initiative, Agency for Science Technology and Research (A*STAR), Singapore, Singapore
gangeshwark@ihpc.a-star.edu.sg

[2] Information Systems Technology and Design, Singapore University of Technology and Design, Singapore, Singapore
{navonil_majumder,soujanya_poria}@sutd.edu.sg

[3] School of Computer Science and Engineering, Nanyang Technological University, Singapore, Singapore
cambria@ntu.edu.sg

Abstract. Automatic deception detection is an important task that has gained momentum in computational linguistics due to its potential applications. In this paper, we propose a simple yet tough to beat multimodal neural model for deception detection. By combining features from different modalities such as video, audio, and text along with Micro-Expression features, we show that detecting deception in real life videos can be more accurate. Experimental results on a dataset of real-life deception videos show that our model outperforms existing techniques for deception detection with an accuracy of 96.14% and ROC-AUC of 0.9799.

Keywords: Automatic deception detection · Multimodal neural model · Expression features

1 Introduction

We face deceptive behavior in our day-to-day life. People lie to escape from a situation that seems unfavorable to them. As a consequence, some lies are innocuous but others may have severe ramifications in the society. Reports suggest that the ability of humans to detect deception without special aids is only 54% [1]. A study by DePaulo et al. [2] found that deception without any particular motivation or intention exhibited almost no detectable cues of deception. However, cues were significantly more when lies were about transgressions.

With the rise in the number of criminal cases filed every year in the US[1], it is ethically and morally important to accuse only the guilty defendant and free the innocent. Since the judgment for any case is mostly based on the hearings and evidence from the stakeholders (accused, witnesses, etc.), the judgment is

[1] http://www.uscourts.gov.

most likely to go wrong if the stakeholders do not speak the truth. It is, hence, important to detect deceptive behavior accurately in order to upkeep the law and order.

Social media can be characterized as a virtual world where people interact with each other without the human feel and touch. It is easy to not reveal one's identity and/or pretend to be someone else on social media [3]. Cyberbullying is increasingly becoming a common problem amongst the teenagers nowadays [4]. These include spreading rumors about a person, threats, and sexual harassment. Cyberbullying adversely affects the victim and leads to a variety of emotional responses such as lowered self-esteem, increased suicidal thoughts, anger, and depression [5–7]. Teenagers fall prey to these attacks due to their inability to comprehend the chicanery and pretentious behavior of the attacker. Another area where deception detection is of paramount importance is with the increased number of false stories, i.e., fake news, on the Internet. Recent reports suggest that the outcome of the U.S. Presidential Elections is due to the rise of online fake news. Propagandists use arguments that, while sometimes convincing, are not necessarily valid. Social media, such as Facebook and Twitter, have become the propellers for this political propaganda. Countries around the world, such as France [8], are employing methods that would prevent the spread of fake news during their elections. Though these measures might help, there is a pressing need for the computational linguistics community to devise efficient methods to fight fake news given that humans are poor at detecting deception.

This paper is organized as follows: Sect. 2 discusses past work in deception detection; Section 3 describes the proposed approach to solving deception detection; Section 4 explains experimental setup; Sections 5 and 6 discuss results and drawbacks, respectively; finally, Sect. 7 concludes the paper and proposes future work.

2 Related Work

Past research in the detection of deception can be broadly classified as verbal and non-verbal. In verbal deception detection, the features are based on the linguistic characteristics, such as n-grams and sentence count statistics [9], of the statement by the subject under consideration. Use of more complex features such as psycholinguistic features [10] based on LIWC, have also been explored by [9] and shown that they are helpful in detecting deceptive behavior. Yancheva and Rudzicz studied the relation between the syntactic complexity of text and deceptive behavior [11]. 1 In non-verbal deception detection, physiological measures were the main source of signals for detecting deceptive behavior. Polygraph tests measure physiological features such as heart rate, respiration rate, skin temperature of the subject under investigation. However, these tests are not reliable and often misleading as indicated by [12,13] since judgment made by humans are often biased. Facial expressions and hand gestures were found to be very helpful in detecting deceptive nature. Ekman [14] defined micro-expressions as short involuntary expressions, which could potentially indicate deceptive behavior. Caso et al. [15] identified particular hand gesture to be important to identify the act of deception. Cohen et al. [16] found that fewer iconic hand gestures were a sign of a deceptive narration.

Previous research was focused on detecting deceit behavior under constrained environments which may not be applicable in real life surroundings. Recently, the focus has been towards experiments in real life scenarios. Towards this, Pérez-Rosas et al. [17] introduced a new multimodal deception dataset having real-life videos of courtroom trials. They demonstrated the use of features from different modalities and the importance of each modality in detecting deception. They also evaluated the performance of humans in deception detection and compared it with their machine learning models. Wu et al. [18] have developed methods that leverage multimodal features for detecting detection. Their method heavily emphasizes on feature engineering along with manual cropping and annotating videos for feature extraction.

In this paper, we describe our attempt to use neural models that uses features from multiple modalities for detecting deception. We believe our work is the first attempt at using neural networks for deceit classification. We show that with the right features and simple models, we can detect deceptive nature in real life trial videos more accurately.

3 Approach

3.1 Multimodal Feature Extraction

The first stage is to extract unimodal features from each video. We extract textual, audio and visual features as described below.

Visual Feature Extraction. For extracting visual features from the videos, we use 3D-CNN [19]. 3D-CNN has achieved state-of-the-art results in object classification on tridimensional data [19]. 3D-CNN not only extracts features from each image frame, but also extracts spatiotemporal features [20,21] from the whole video which helps in identifying the facial expressions such as smile, fear, or stress.

The input to 3D-CNN is a video v of dimension (c, f, h, w), where c represents the number of channels and f, h, w are the number of frames, height, and width of each frames respectively. A 3D convolutional filter, f_l of dimension (f_m, c, f_d, f_h, f_w) is applied, where f_m = number of feature maps, c = number of channels, f_d = number of frames (also called depth of the filter), f_h = height of the filter, and f_w = width of the filter. This filter, f_l, produces an output, *convout* of dimension $(f_m, c, f - f_d + 1, h - f_h + 1, w - f_w + 1)$ after sliding across the input video, v. Max pooling is applied to *convout* with window size being (m_p, m_p, m_p). Subsequently, this output is fed to a dense layer of size d_f and softmax. The activations of this dense layer is used as the visual feature representation of the input video, v.

In our experiments, we consider only RBG channel images, hence $c = 3$. We use 32 feature maps and 3D filters of size, $f_d = f_h = f_w = 5$. Hence, the dimension of the filter, f_l, is $32 \times 3 \times 5 \times 5 \times 5$. The window size, m_p, of the max pooling layer is 3. Finally, we obtain a feature vector, v_f, of dimension 300 for an input video, v.

Textual Features Extraction. In order to extract features from the transcript of a video, v, we use Convolutional Neural Networks (CNN) [22,23]. First, we use pre-trained Word2Vec [24] model to extract the vector representations for every word in the transcript. These vectors are concatenated and fed as input vector to the CNN. We use a simple CNN with one convolutional layer and a max-pooling layer, to get our sentence representation. In our experiments, filters of size 3, 5 and 8 with 20 feature maps each is used. Window-size of 2 is employed for max-pooling over these feature maps. Subsequently, a full-connected layer with 300 neurons is used with rectified linear unit (ReLU) [25] as the activation function. The activations of this full-connected layer is used as the textual feature representation of the input video, v. Finally, we obtain a feature vector, t_f, of dimension 300 for an input text (transcript), t.

Audio Feature Extraction. openSMILE [26] is an open-source toolkit used to extract high dimensional features from an audio file. In this work, we use openSMILE to extract features from the input audio. Before extracting the features, we make sure that there are no unnecessary signals in the audio that affects the quality of the extracted features. Hence, the background noise is removed from the audio and Z-standardization is used to perform voice normalization. To remove the background noise, we use SoX (Sound eXchange) [27] audio processing tool. The noiseless input audio is then fed to the openSMILE tool to extract high-dimensional features. These features are functions of low-level descriptor (LLD) contours. Specifically, we use the *IS13-ComParE* openSMILE configuration to extract features which are of dimension 6373 for every input audio, a. After these features are extracted, a simple fully-connected neural network is trained to reduce the dimension to 300. Finally, we obtain a feature vector, a_f, of dimension 300 for an input audio, a.

Micro-Expression Features. Veronica et al. manually annotated facial expressions and use binary features derived from the ground truth annotations to predict deceptive behavior. Facial micro-expressions are also considered to play an important role in detecting deceptive behavior. The data provided by [17] contains 39 facial micro-expressions such as frowning, smiling, eyebrows raising, etc. These are binary features and taken as a feature vector, m_p of dimension 39.

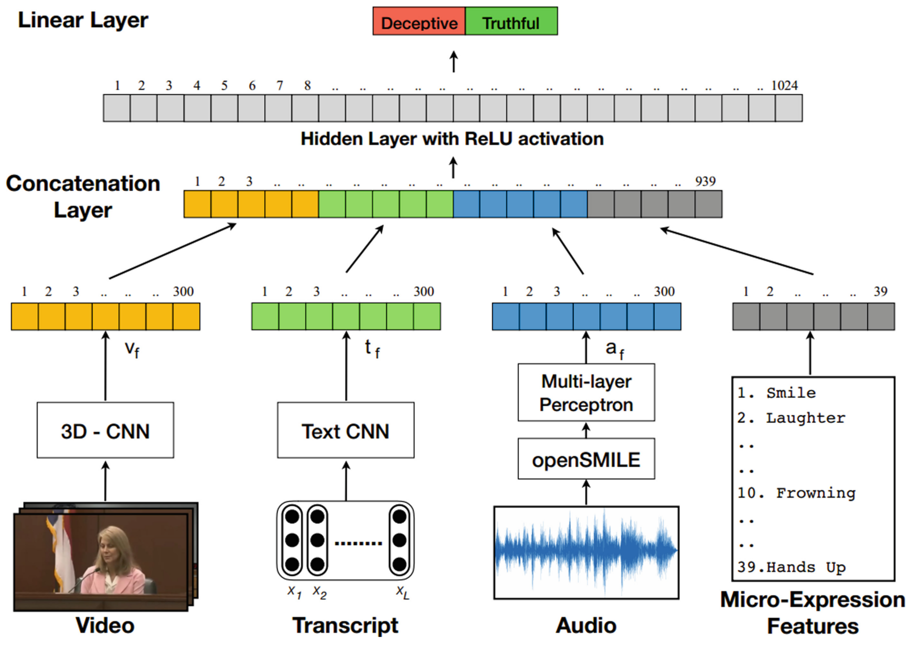


Fig. 1. Architecture of model MLP_C

3.2 Model Description

Multimodal Model. We use a simple Multi-Layer perceptron (MLP) with hidden layer of size 1024 followed by a linear output layer. We use the rectified linear unit (ReLU) activation function [25] for non-linearity at the hidden layer. A dropout [28] of keep probability, $p = 0.5$, is applied to the hidden layer for regularization.

Figures 1 and 2 shows the architecture of our models, MLP_C and MLP_{H+C}.

Unimodal Models. We perform the evaluation on individual modalities and use the same architecture as multimodal model. The only difference is that the input is either v_f, or a_f, or t_f, or m_f. Hence no data fusion is performed. We name the model as MLP_U.

3.3 Data Fusion

We fuse the features from individual modalities to map them into a joint space. To achieve this, we try different kinds of data fusion techniques:

Concatenation. In this method, the features from all the modalities are simply concatenated into a single feature vector. Thus, the extracted features, t_f, a_f, v_f

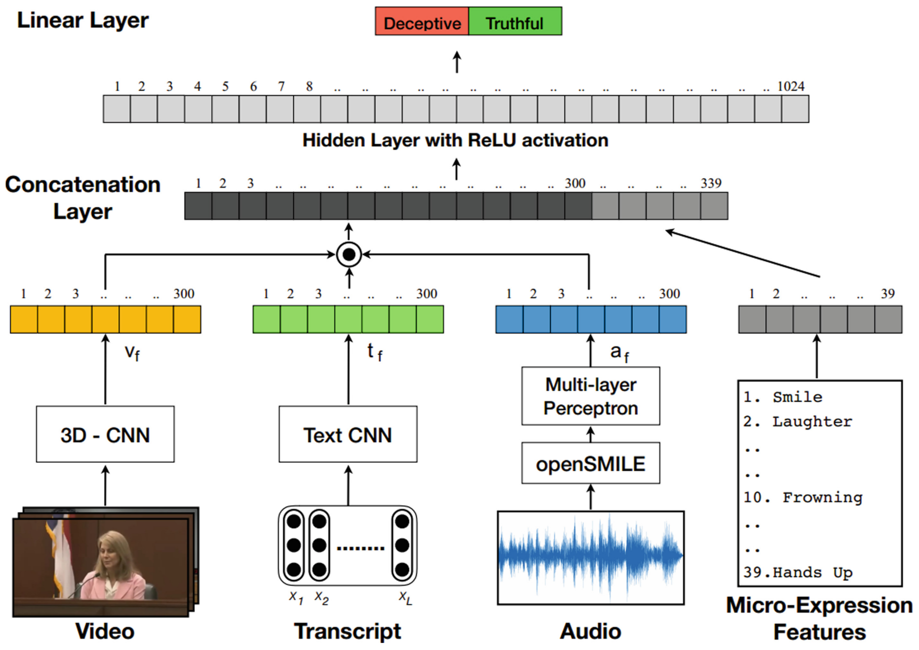


Fig. 2. Architecture of model MLP_{H+C}

and m_f, are simply concatenated to form the representation: $z_f = [t_f; a_f; v_f; m_f]$ of dimension $d_{in} = 939$.

We call this model configuration as MLP_C and is shown in the Fig. 1.

Hadamard + Concatenation. In this method, the audio features, visual features and, textual features are fusion by using hadamard product. Then the Micro-Expression features are concatenated with the product. Thus, we have $z_f = [t_f \odot a_f \odot v_f; m_f]$ of dimension $d_{in} = 339$, where $(A \odot B)$ is element-wise multiplication between matrices A and B (also known as Hadamard product).

We call this model configuration as MLP_{H+C} and is shown in the Fig. 2.

3.4 Loss

The training objective is to minimize the cross-entropy between the model's output and the true labels. We trained the models with back-propagation using Stochastic Gradient Descent optimizer. The loss function is:

$$J = \frac{-1}{N} \sum_{i=1}^{N} \sum_{j=1}^{C} y_{i,j} \; log_2(\hat{y}_{i,j}) \tag{1}$$

Here, N is the number of samples, C is the number of categories (in our case, $C = 2$). y_i is the one-hot vector ground truth of i^{th} sample and $\hat{y}_{i,j}$ is its predicted probability of belonging to class j.

4 Experiment

4.1 Data

For evaluating our deception detection model, we use a real-life deception detection dataset by [17]. This dataset contains 121 video clips of courtroom trials. Out of these 121 videos, 61 of them are of deceptive nature while the remaining 60 are of truthful nature.

The dataset contains multiple videos from one subject. In order to avoid bleeding of personalities between train and test set, we perform a 10-fold cross validation with subjects instead of videos as suggested by Wu et al. [18]. This ensures that videos of the same subjects are not in both training and test set.

4.2 Baselines

Wu et al. [18] have made use of various classifiers such as Logistic Regression (LR), Linear SVM (L-SVM), Kernel SVM (K-SVM). They report the AUC-ROC values obtained by the classifiers for different combination of modalities. We compare the AUC obtained by our models against only Linear SVM (L-SVM) and Logistic Regression (LR).

Pérez-Rosas et al. [17] use Decision Trees (DT) and Random Forest (RF) as their classifiers. We compare the accuracies of our models against DT and RF.

Table 1. Comparison of AUC

Features	L-SVM [18]	LR [18]	MLP_U	MLP_C	MLP_{H+C}
Random	–	–	0.4577	0.4788	0.4989
Audio	**0.7694**	0.6683	0.5231	–	–
Visual	0.7731	0.6425	**0.9596**	–	–
Textual (static)	0.6457	0.5643	**0.8231**	–	–
Textual (non-static)	–	–	**0.9455**	–	–
Micro-expression	0.7964	**0.8275**	0.7512	–	–
All features (static)	0.9065	0.9221	–	0.9033	**0.9348**
All features (non-static)	–	–	–	0.9538	**0.9799**

5 Results

Tables 1 and 2 presents the performances of MLP and its variants along with the state-of-the-art models. During feature extraction from text, we train the TextCNN model with two different settings: one, by keeping the word vector representation static; two, by optimizing the vector along with the training (non-static). In our results, we also show the performance of the model with these two textual features separately. Additionally, we also mention the results we got from our models for feature vectors initialized with random numbers.

Table 1 shows that our model, MLP_{H+C}, obtains an AUC of 0.9799 while outperforming all other competitive baselines with a huge margin.

Table 2. Comparing accuracies of our model with baselines

Features	DT [17]	RF [17]	MLP_U	MLP_C	MLP_{H+C}
Random	–	–	43.90%	45.32%	48.51%
Audio	–	–	**52.38%**	–	–
Visual	–	–	**93.08%**	–	–
Textual (Static)	60.33%	50.41%	**80.16%**	–	–
Textual (Non-static)	–	–	**90.24%**	–	–
Micro-Expression	68.59%	73.55%	**76.19%**	–	–
All Features (Static)	75.20%	50.41%	-	90.49%	**90.99%**
All Features (Non-static)	–	–	–	95.24%	**96.14%**

Table 2 compares the performance our models with Decision Tree and Linear Regression models [17]. Our model, MLP_{H+C}, again outperforms other baselines by achieving an accuracy of 96.14%. We can also infer that visual and textual features play a major role in the performance of our models; followed by Micro-Expressions and audio. This conforms with the findings by [17] that facial display features and unigrams contribute the most to detecting deception.

As we can see that, our approach outperforms the baselines by a huge margin. Our neural models simple and straightforward, hence the results show that right feature extraction methods can help in unveiling significant signals that are useful for detecting deceptive nature.

6 Drawbacks

Though our models outperform the previous state-of-the-art models, we still acknowledge the drawbacks of our approach as follows:

– Our models still rely on a small dataset with only 121 videos. Due to this, our models are prone to over-fitting if not carefully trained with proper regularization.
– Also, due to the limited scenarios in the dataset, the model may not show the same performance for out-of-domain scenarios.

7 Conclusions and Future Work

In this paper, we presented a system to detect deceptive nature from videos. Surprisingly, our model performed well even with only 121 videos provided in the dataset, which is generally not a feasible number of data points for neural models. As a result, we conclude that there exists a certain pattern in the videos that provide highly important signals for such precise classification. We performed various other evaluations not presented in this paper, to confirm the performance of our model. From these experiments, we observed that visual and textual features predominantly contributed to accurate predictions followed by Micro-Expression features. Empirically, we observed that our model MLP_{H+C} converged faster in comparison with MLP_C.

While our system performs well on the dataset by Pérez-Rosas et al. [17], we can not claim the same performance of our model for larger datasets covering a larger number of environments into consideration. Hence, creating a large multimodal dataset with a large number of subjects under various environmental condition is part of our future work. This would pave a way to build more robust and efficient learning systems for deception detection. Another interesting path to explore is detecting deception under social dyadic conversational setting.

References

1. Bond, C.F., Jr., DePaulo, B.M.: Accuracy of deception judgments. Pers. Soc. Psychol. Rev. **10**, 214–234 (2006)
2. DePaulo, B.M., Lindsay, J.J., Malone, B.E., Muhlenbruck, L., Charlton, K., Cooper, H.: Cues to deception. Psychol. Bull. **129**, 74 (2003)
3. Cambria, E., Wang, H., White, B.: Guest editorial: big social data analysis. Knowl.-Based Syst. **69**, 1–2 (2014)
4. Smith, P.K., Mahdavi, J., Carvalho, M., Fisher, S., Russell, S., Tippett, N.: Cyberbullying: its nature and impact in secondary school pupils. J. Child Psychol. Psychiatry **49**, 376–385 (2008)
5. Ji, S., Pan, S., Li, X., Cambria, E., Long, G., Huang, Z.: Suicidal ideation detection: a review of machine learning methods and applications. IEEE Trans. Comput. Soc. Syst. **8**, 214–226 (2021)
6. Ji, S., Li, X., Huang, Z., Cambria, E.: Suicidal ideation and mental disorder detection with attentive relation networks. Neural Comput. Appl. **34**, 10309–10319 (2022)
7. Hinduja, S., Patchin, J.W.: Bullying Beyond the Schoolyard: Preventing and Responding to Cyberbullying. Corwin Press, Thousand Oaks (2014)
8. Baptiste Su, J.: France to impose restrictions on Facebook, Twitter in fight against fake news during elections (2018). [Online; posted 09-January-2018]
9. Mihalcea, R., Pulman, S.: Linguistic ethnography: identifying dominant word classes in text. In: Gelbukh, A. (ed.) CICLing 2009. LNCS, vol. 5449, pp. 594–602. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-00382-0_48
10. Pennebaker, J.W., Francis, M.E., Booth, R.J.: Linguistic Inquiry and Word Count: LIWC 2001, p. 71. Lawrence Erlbaum Associates, Mahway (2001)

11. Yancheva, M., Rudzicz, F.: Automatic detection of deception in child-produced speech using syntactic complexity features. In: Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 944–953. Association for Computational Linguistics (2013)
12. Vrij, A.: Detecting Lies and Deceit: The Psychology of Lying and Implications for Professional Practice. Wiley, Hoboken (2000)
13. Gannon, T.A., Beech, A.R., Ward, T.: Risk assessment and the polygraph. The use of the polygraph in assessing, treating and supervising sex offenders: a practitioner's guide, pp. 129–154 (2009)
14. Ekman, P.: Telling Lies: Clues to Deceit in the Marketplace, Politics, and Marriage, Revised edn. WW Norton & Company, New York (2009)
15. Caso, L., Maricchiolo, F., Bonaiuto, M., Vrij, A., Mann, S.: The impact of deception and suspicion on different hand movements. J. Nonverbal Behav. **30**, 1–19 (2006)
16. Cohen, D., Beattie, G., Shovelton, H.: Nonverbal indicators of deception: how iconic gestures reveal thoughts that cannot be suppressed. Semiotica **2010**, 133–174 (2010)
17. Pérez-Rosas, V., Abouelenien, M., Mihalcea, R., Burzo, M.: Deception detection using real-life trial data. In: Proceedings of the 2015 ACM on International Conference on Multimodal Interaction, pp. 59–66. ACM (2015)
18. Wu, Z., Singh, B., Davis, L.S., Subrahmanian, V.: Deception detection in videos. arXiv preprint arXiv:1712.04415 (2017)
19. Ji, S., Xu, W., Yang, M., Yu, K.: 3D convolutional neural networks for human action recognition. IEEE Trans. Pattern Anal. Mach. Intell. **35**, 221–231 (2013)
20. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 4489–4497 (2015)
21. Chaturvedi, I., Satapathy, R., Cavallari, S., Cambria, E.: Fuzzy commonsense reasoning for multimodal sentiment analysis. Pattern Recogn. Lett. **125**, 264–270 (2019)
22. Kim, Y.: Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882 (2014)
23. Kalchbrenner, N., Grefenstette, E., Blunsom, P.: A convolutional neural network for modelling sentences. arXiv preprint arXiv:1404.2188 (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. Nair, V., Hinton, G.E.: Rectified linear units improve restricted Boltzmann machines. In: Proceedings of the 27th International Conference on Machine Learning (ICML-2010), pp. 807–814 (2010)
26. Eyben, F., Weninger, F., Gross, F., Schuller, B.: Recent developments in opensmile, the Munich open-source multimedia feature extractor. In: Proceedings of the 21st ACM International Conference on Multimedia. MM 2013, pp. 835–838. ACM, New York (2013)
27. Norskog, L.: Sound exchange (1991). http://sox.sourceforge.net/
28. 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**, 1929–1958 (2014)

Author Profiling and Authorship Attribution, Social Network Analysis

Hamtajoo: A Persian Plagiarism Checker for Academic Manuscripts

Vahid Zarrabi(✉), Salar Mohtaj, and Habibollah Asghari

ICT Research Institute, Academic Center for Education, Culture and Research (ACECR), Tehran, Iran
{vahid.zarrabi,salar.mohtaj,habib.asghari}@ictrc.ac.ir

Abstract. In recent years, due to the high availability of electronic documents through the Web, the plagiarism has become a serious challenge, especially among scholars. Various plagiarism detection systems have been developed to prevent text re-use and to confront plagiarism. Although it is almost easy to detect duplicate text in academic manuscripts, finding patterns of text re-use that has been semantically changed is of great importance. Another important issue is to deal with less resourced languages, which there are low volume of text for training purposes and also low performance in tools for NLP applications. In this paper, we introduce Hamtajoo, a Persian plagiarism detection system for academic manuscripts. Moreover, we describe the overall structure of the system along with the algorithms used in each stage. In order to evaluate the performance of the proposed system, we used a plagiarism detection corpus comply with the PAN standards.

Keywords: Plagiarism detection · Text re-use · Text similarity

1 Introduction

Plagiarism refers to the unacknowledged use of others' text or ideas. Stopping plagiarism is an important topic in the field of academic publication. Plagiarism detection systems compare a submitted suspicious document against a collection of source documents to find the cases of text re-use. It has been proved that plagiarism detection systems are effective tools for discouraging researchers to commit plagiarism [1].

There are two main categories for plagiarism detection (PD) systems. External plagiarism detection systems search for pattern of text re-use between suspicious document and a collection of source documents. On the other hand, intrinsic plagiarism detection systems exploit stylometry algorithms to find the changes in writing style of the author.

There are different plagiarism detection tools that have been investigated in multiple surveys [2,3]. Crosscheck [4,5] and Turnitin [6] are the most popular ones in academic community. Although there are many plagiarism detection systems in English, plagiarism detection is still a serious task in less resourced languages.

https://doi.org/10.1007/978-3-031-23793-5_9

In this paper, we propose *Hamtajoo*, a Persian plagiarism detection framework for investigating patterns of text re-use in Persian academic papers. Persian belongs to Arabic script-based languages which share some common characteristics. This language family has some common properties such as absence of capitalization, right to left direction, encoding issues in computer environment and lack of clear word boundaries in multi-token words [7]. The system works on a document level at the first stage and then focuses of paragraph and sentence level in the second detailed comparison stage. We have prepared a reference collection of all of the Persian academic papers indexed by SID[1] (Scientific Information Database).

The rest of the paper is organized as follows. Section 2 describes the related work on recent plagiarism detection tools and methods. In Sect. 3, we explain the proposed approach and also describe the algorithms that have been developed and implemented in the system. Section 4 comes with system design, architecture and functionalities. In Sect. 5, an evaluation framework for examining the system is described. Conclusion and the directions for the future work are presented in the final section.

2 Related Work

In this section we investigate some of the available plagiarism detection systems.

An Arabic plagiarism detection tool called Aplag has been introduced in [8]. For developing the plagiarism checker system, they have extracted the fingerprints on document, paragraph and sentence levels for saving the computation time. If the similarity between hashes of the two documents is above a specific threshold, then the process continues to paragraph level and so on.

In a work accomplished by Alzahrani et al., an intelligent plagiarism reasoned named as iPlag, has been designed [9]. Scientific publications in same fields usually share same general information and have some common knowledge. Besides, each publication should convey specific contributions. In this system, they have processed various parts of a manuscript and weighted them based on their importance in plagiarism detection. So, the PD system pays more attention to the parts of a manuscript that has more contributions and lower weights given to less important parts.

Meuschke et al. proposed Citeplag, a plagiarism detection system based on citation pattern matching [10]. They search similar patterns of citations between source and suspicious documents to find cases of plagiarism.

In a work accomplished by Leong and Lau, they have developed a document plagiarism detection system named as Check [11]. They try to eliminate unnecessary comparisons between documents with different subjects and so reduced the computational costs.

A word-similarity sentence-based plagiarism detection tool on Web documents, named as SimPaD has been developed in [12]. They measure the similarity

[1] http://www.sid.ir/.

between sentences by computing word correlation factors and then generate a graphical view of sentences that are similar.

Collberg et al. developed a system for self-plagiarism detection named as SPLAT [13]. The system crawls websites of top fifty computer science departments and downloads the research papers. Then a text comparison algorithm compares all of the papers for instances of text re-use.

It should be noted that most of the tools for detection cases of plagiarism are only pay attention to instances of verbatim copy plagiarism and cannot identify paraphrased passages of text which require semantic similarity detection methods. Our contribution in this paper is to use specific features in Persian to detect cases of paraphrased plagiarism.

3 Proposed Approach

The proposed approach for developing *Hamtajoo* system is thoroughly described in this section.

Stein et al. proposed a generic three-step retrieval process for an external PD system [14] that is depicted in Fig. 1. Their proposed model consists of a heuristic retrieval step for extracting a set of source documents from the source collection, a detailed analysis step for comparing the suspicious document with candidate source documents for finding the similar passages, and a knowledge-based post-processing step to analyze identical passages and investigate whether they contain proper quotations.

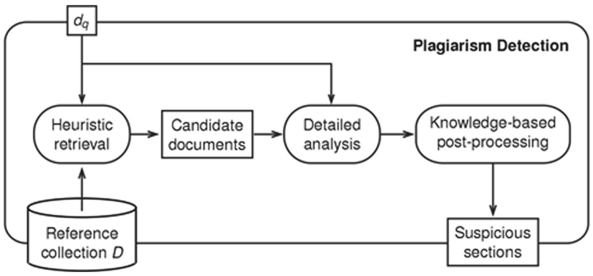


Fig. 1. Generic three-step retrieval process for an external plagiarism detection system [14]

A similar approach has been used to develop *Hamtajoo* PD system. *Hamtajoo* includes two main components, including the candidate retrieval and the text alignment modules. From the evaluation point of view, candidate retrieval is a recall-oriented task, while precision is mainly focused by the subsequent text alignment step [15].

In this section we describe main proposed methods for the candidate retrieval and the text alignment modules. The evaluation results on the proposed algorithms are presented in the next section.

3.1 Candidate Retrieval Module

Since the comparison of submitted suspicious document with the entire source documents in the system would be very time consuming, a candidate retrieval stage is considered to decrease the search space. The aim is to find most similar documents with the submitted suspicious document and also to reduce the number of documents for the subsequent text alignment stage. Our approach to retrieve candidate documents is divided into four main steps:

- Chunking the suspicious document
- Noun phrase and keyword phrase extraction
- Query formulation
- Search control

Before these main steps, suspicious documents are passed through Parsiver pre-processing [16] block that includes stop words removal and unification of punctuation marks. Parsiver is an integrated package written in python which performs different kinds of pre-processing tasks in Persian. Each of the mentioned steps is described in more details as follows.

Chunking the Suspicious Document: In this step, the submitted suspicious document is segmented into some parts called chunks. These chunks will be used for query construction based on keyword phrase and noun phrase extraction. Therefore, their length should be long enough to extract meaningful queries. On the other hand, these chunks may contain unknown numbers of plagiarism cases from source documents. Based on experiments on *Hamtajoo* to choose the system parameters, 500 words length has been chosen as the chunk length. In other words, the suspicious document would be divided into chunks of 500 words length and then each chunk is tokenized into individual sentences.

Noun Phrase and Keyword Phrase Extraction: This step has the main role in candidate retrieval task. In other words, extracting appropriate keywords could lead the candidate retrieval to perform more efficiently. There are many previous studies that tried to extract keywords by investigating content [17,18]. In order to construct queries from the suspicious document in our approach, both keyword phrases and noun phrases are extracted.

Before starting the extraction process, sentences with low information content are discarded. For this purpose, the input sentences have been ranked based on their length and the number of nouns, and then we discard the lower 20% of the sentences. The resulting sentences are long enough and have rich content for keyword extraction. The TF-IDF (Term Frequency - Invert Document Frequency) weighting scheme has been used to extract important words from the sentences. The keywords are extracted from top 3 most important sentences that contain the highest TF-IDF words. The chosen keywords include nouns with high TF-IDF values, remaining nouns in the sentence, and also adjectives and verbs with high TF-IDF values. Moreover, the noun phrase extraction is accomplished by processing the remaining sentences. The formulation is deployed based on the formal Persian noun phrase structure. For each noun phrase, a score is calculated based on TF-IDF values.

Query Formulation: For top ranked sentences selected from previous step, the extracted keywords are simply placed next to each other based on their order in sentence and are passed to the next step as a query. The Apache Lucene[2] is used to index the source documents. The constructed queries in this section are passed to Lucene application programming interface (API) to search them within the indexed documents.

Search Control: In this step, some of the constructed queries are dropped based on the previously downloaded documents. The input query is compared against the downloaded documents that are gathered from previous rounds of source retrieval steps.

The retrieved documents based on the constructed queries are passed to the text alignment sub-system for more detailed analysis and comparison. For each submitted documents, 25 source documents are chosen in the average for text alignment analysis in the next stage.

3.2 Text Alignment Module

The candidate documents from the previous stage are passed to a text alignment module for detail comparison of text passages between suspicious document and the source candidates. In this stage, the exact position of text re-use cases will be detected and will be reported to the end users. Different methods and algorithms have been tested to choose the best one from the accuracy and speed points of view. A standard PD corpus and also a plagiarism detection lab are designed to compare the performance of different methods. The detail description of the compiled corpus will be presented in the "Experiments and Evaluation" section.

Figure 2 shows a snapshot of our PD lab. As depicted in the figure, different methods including character n-gram, word n-gram, Vector Space Model (VSM) and Latent Semantic Analysis (LSA) are deployed in the tool. Moreover, different parameters such as the range of n for n-gram similarity, the similarity threshold for VSM and LSA and the parameters related to different parts of pre-processing are embedded in our PD lab to analyze their impact on plagiarism detection performance.

As depicted in the figure, the PD lab includes a dot plot graph that shows the cases of similarity between pairs of documents. Among different methods which have been developed in our PD lab, the VSM similarity is chosen based on accuracy and runtime criteria.

The proposed model for detecting exact position of text re-use cases includes the following steps; First, we split the pairs of suspicious and candidate source documents into sentences. In the second step, for each sentence in source and suspicious document, the relevant vectors have been created based on TF-IDF weighting schema. The IDF measure has been computed on a large collection of academic manuscript and the TF counts the number of target word in the

[2] https://lucene.apache.org/.

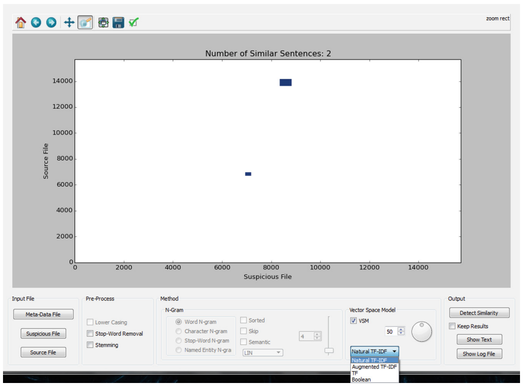


Fig. 2. Snapshot of plagiarism detection lab

document. Finally, in the third step, a pairwise cosine similarity has been computed between all of the sentences in two documents. Pairs of the sentences with similarity higher than predefined threshold are considered as cases of text re-use.

4 System Architecture

In this section, we describe the main technologies that are used to develop the PD system. *Hamtajoo* is a web application composed of two main subsystems; the core (back-end) and front-end subsystems. The main stages of *Hamtajoo* are depicted in Fig. 3.

4.1 Core (Back-end) Sub-system

The Django (1.8) web framework has been used to develop the core subsystem of *Hamtajoo*. Django is a free and open-source web framework, written in Python, which follows the model-view-template (MVT) architectural pattern. The user authentication, raw text extraction from submitted documents and the candidate retrieval and text alignment tasks are done in the core subsystem.

To make a standard format from both submitted suspicious document and the source documents in the system, the Parsiver pre-processing toolkit [16] is used. Parsiver has been used to normalize all of the character encodings into a unified format and also to tokenize words in text documents. Moreover, we used Parsiver stemmer for different functions in text alignment module of the system.

The input documents to *Hamtajoo* can be of various file formats including ".doc", ".docx" and ".txt" formats. To extract the raw text from .doc and .docx documents, the win32com.client and docx2txt modules are used, respectively. The docx2txt is a pure python-based utility to extract text from .docx files. The code is taken and adapted from python-docx[3]. We have used these modules to extract the body of text from the submitted documents.

[3] https://github.com/ankushshah89/python-docx2txt.

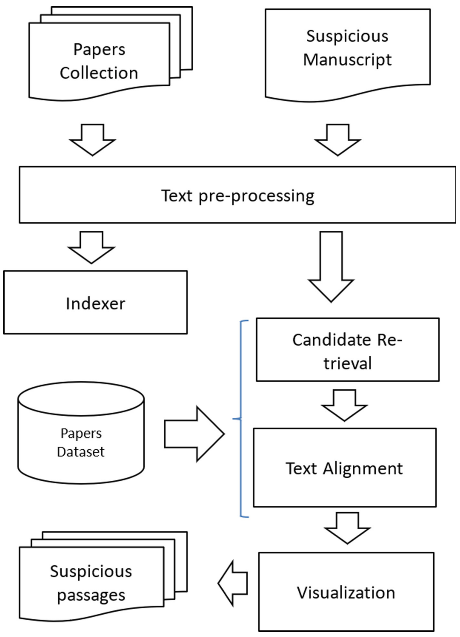


Fig. 3. The block diagram of *Hamtajoo* PD system

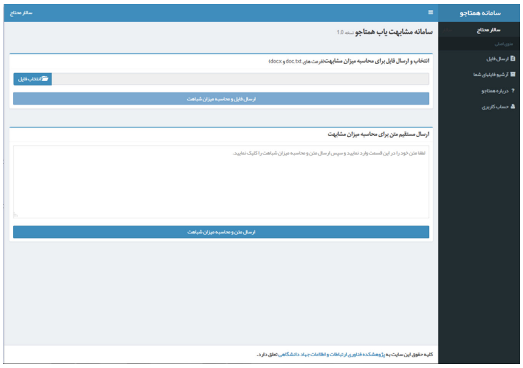
In order to construct the collection of source documents, all of the papers from SID scientific database were fed into the *Hamtajoo* system. To do so, the Apache Lucene platform has been used to index the text documents. While the python is used to develop the core subsystem, the PyLucene API is used to develop the indexing system. PyLucene is a Python extension for accessing Java Lucene. Its goal is to allow using Lucene text indexing and searching capabilities from Python[4]. Moreover, MySQL database is employed to store the metadata information (e.g. author, title and publishing year of paper) that are extracted from indexed papers in the system.

4.2 Front-end Sub-system

The front-end subsystem contains user interface which makes it possible for end users to use *Hamtajoo* for investigating plagiarism in their documents. Moreover, users can create and manage user accounts using the system interface. We exploited the Bootstrap web framework to develop the front-end subsystem. Bootstrap is a free and open-source front-end Web framework for designing websites and Web applications. It contains HTML- and CSS-based design templates for typography, forms, buttons, navigation and other interface components, as well as optional JavaScript extensions.

[4] https://lucene.apache.org/pylucene/index.html.

Figure 4 shows the main parts of front-end section of *Hamtajoo*. This figure shows the main page of the system, where users can submit their documents for plagiarism analysis. It is possible for users to submit the raw text directly into the system and to upload their documents in doc, docx and txt formats.


سامانه همتاجو
سالار محتاج
سامانه مشابهت یاب همتاجو نسخه 1.0
منوی اصلی
ارسال فایل
آرشیو فایلهای شما
درباره همتاجو
حساب کاربری
انتخاب و ارسال فایل برای محاسبه میزان مشابهت(فرمت های doc.txt و docx)
انتخاب فایل
ارسال فایل و محاسبه میزان شباهت
ارسال مستقیم متن برای محاسبه میزان مشابهت
ارسال متن و محاسبه میزان شباهت

لطفا متن خود را در این قسمت وارد نمایید و سپس ارسال متن و محاسبه میزان شباهت را کلیک نمایید.

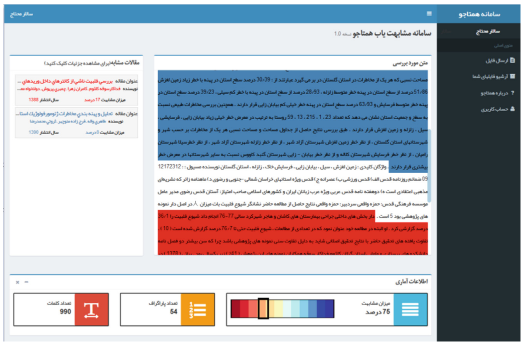
کلیه حقوق این سایت به پژوهشکده فناوری ارتباطات و اطلاعات جهاد دانشگاهی تعلق دارد.

Fig. 4. Manuscript submission page of *Hamtajoo*

Figure 5 shows the system output for a submitted suspicious document. The system highlights the section of the submitted text which contains cases of text re-use with different colors for different source papers. Moreover, some general statistical information related to submitted document (i.e. number of words, number of paragraphs and ratio of plagiarism in the document) are depicted in the bottom side of the page. List of the source papers which have cases of text similarity with the submitted document can be shown in system output (two papers in this example). Users can examine the detail text similarity between the submitted text and source papers by clicking on listed papers.

Table 1. Overall detection performance of *Hamtajoo* vs. the algorithms presented in Persian PlagDet 2016 [20]

Team	Recall	Precision	Granularity	F-Measure	Plagdet
Hamtajoo	*0.9221*	*0.9345*	*1*	*0.9282*	*0.9282*
Mashhadirajab	0.9191	0.9268	1.0014	0.9230	0.9220
Gharavi	0.8582	0.9592	1	0.9059	0.9059
Momtaz	0.8504	0.8925	1	0.8710	0.8710
Minaei	0.7960	0.9203	1.0396	0.8536	0.8301
Esteki	0.7012	0.9333	1	0.8008	0.8008
Talebpour	0.8361	0.9638	1.2275	0.8954	0.7749
Ehsan	0.7049	0.7496	1	0.7266	0.7266
Gillam	0.4140	0.7548	1.5280	0.5347	0.3996
Mansourizadeh	0.8065	0.9000	3.5369	0.8507	0.3899


سامانه همتاجو

سالار محتاج

سامانه مشابهت یاب همتاجو نسخه 1.0

ارسال فایل
آرشیو فایلهای شما
درباره همتاجو
حساب کاربری

متن مورد بررسی

مساحت نسبی که هر یک از مخاطرات در استان گلستان در بر می گیرد عبارتند از : 30/39 درصد سطح استان در پهنه با خطر زیاد زمین لغزش 51/86 درصد از سطح استان در پهنه خطر متوسط زلزله ، 28/93 درصد از سطح استان در پهنه با خطر کم سیلی ، 39/23 درصد سطح استان در پهنه خطر متوسط فرسایش و 63/93 درصد سطح استان در پهنه خطر خیلی کم بیابان زایی قرار دارند . همچنین بررسی مخاطرات طبیعی نسبت به سطح و جمعیت استان نشان می دهد که تعداد 23 ، 1 ، 215 ، 13 ، 59 روستا به ترتیب در معرض خطر خیلی زیاد بیابان زایی ، فرسایش ، سیل ، زلزله و زمین لغزش قرار دارند . طبق بررسی نتایج حاصل از جداول مساحت و مساحت نسبی هر یک از مخاطرات بر حسب شهر و شهرستانهای استان گلستان ، از نظر خطر زمین لغزش شهرستان آزاد شهر ، از نظر خطر زلزله شهرستان آزاد شهر ، از نظر خطرسیلا شهرستان رامیان ، از نظر خطر فرسایش شهرستان کلاله و از نظر خطر بیابان – زایی شهرستان گنبد کاووس نسبت به سایر شهرستانها در معرض خطر بیشتری قرار دارند . واژگان کلیدی : زمین لغزش ، سیل ، بیابان زایی ، فرسایش خاک ، زلزله ، استان گلستان نویسنده مسیول : : 12172312 09 ضمائم روز نامه قدس الف) قدس ورزشی ب) عصرانه ج) قدس ویژه استانهای خراسان شمالی -جنوبی و رضوی د) ماهنامه زائر که نشریهای مذهبی اعتقادی است ه) دوهفته نامه قدس عربی ویژه عرب زبانان ایران و کشورهای اسلامی صاحب امتیاز: آستان قدس رضوی مدیر عامل موسسه فرهنگی قدس: حمزه واقعی سردبیر: حمزه واقعی نتایج حاصل از مطالعه حاضر نشانگر شیوع فلبیت بات میزان ٪۸ در اصل دار نمونه های پژوهشی بود 5 است . دار بخش های داخلی جراحی بیمارستان های کاشان و هاجر شهرکرد سالی 77-76 انجام داد شیوع فلبیت را 36/1 درصد گزارشی کرد . او البته در مطالعه خود عنوان نمود که در تعدادی از مطالعات ، شیوع فلبیت حتی تا 76/7 درصد گزارش شده است (10) . تفاوت یافته های تحقیق حاضر با نتایج تحقیق اصلانی شاید به دلیل تفاوت سنی نمونه های پژوهشی باشد چرا که سن بیشتر دو فصل نامه

مقالات مشابه(برای مشاهده جزئیات کلیک کنید)

عنوان مقاله بررسي فلبيت ناشي از كاتترهاي داخل وريدهاي ...
نویسنده فداكارسوقه كلثوم. كامران زهرا. چمبري پريوش. دولتخواه مه...
میزان مشابهت 17 درصد سال انتشار 1388

عنوان مقاله تحليل و پهنه بندي مخاطرات ژئومورفولوژيك استا...
نویسنده طاهري واله. فرج زاده منوچهر. ثروتي محمدرضا
میزان مشابهت 8 درصد سال انتشار 1390

اطلاعات آماری

میزان مشابهت 75 درصد

تعداد پاراگراف 54

تعداد کلمات 990

Fig. 5. System output for detail comparison of source and suspicious documents

5 Experiments and Evaluation

In order to evaluate *Hamtajoo* plagiarism detection system, two different experiments have been accomplished to measure the performance of different subsystems. Our candidate retrieval module has been evaluated in PAN 2015 international competition on plagiarism detection [19]. The results are presented in the following subsection. To evaluate the text alignment module, the standard PD dataset of Persian PlagDet shared tasks [20] on plagiarism detection has been used. The performance of text alignment module is compared with all of the proposed methods in Persian PlagDet competition.

5.1 Candidate Retrieval Evaluation

As mentioned in the previous section, in order to evaluate the proposed candidate retrieval approach, we participated in the source retrieval task of PAN 2015 international shared task on plagiarism detection. The candidate retrieval module of *Hamtajoo* achieved the best results in "runtime" and "No detection" measures. Moreover, *Hamtajoo* achieved the second rank in recall measure and also the number of queries among all of the participants. The results of source retrieval shared task are presented in detail in [19].

5.2 Text Alignment Evaluation

The Persian PlagDet shared task at PAN 2016 [20] has been organized to promote the comparative assessment of NLP techniques for plagiarism detection with a special focus on plagiarism that appears in a Persian text corpus. Since the shard task was focused on Persian, we have evaluated the performance of *Hamtajoo* using standard Persian PlagDet 2016 evaluation corpus. Table 1 shows the performance results of *Hamtajoo* in comparison to participants of Persian PlagDet 2016. As mentioned in the table, *Hamtajoo* outperforms other systems in different evaluation measures.

6 Conclusion and Future Works

In this paper, we introduced *Hamtajoo*, a Persian plagiarism detection system. It has been built around a semantic-based method considering specific features of Persian language. The system contains a resource collection comprised of about 480000 journal papers in Persian. Pre-processing of text documents was done in order to transform them into a unified representation, normalizing the text (e.g. unification of various character encodings) and sentence/word tokenization.

By exploiting a graph showing the distribution of plagiarized passages across the document, the expert could achieve a better view of re-used text. The experimental results show the effectiveness of *Hamtajoo* and its competitiveness against the other plagiarism checker tools. For adapting the system to a commercial package, a web-based system is developed to present the system to the academic society.

As a plan for the future works, we aim to conduct more experiments on larger texts to detect the text re-use in long documents such as theses and dissertations with a reasonable computational complexity. Another additional work is to deal with bilingual algorithms for detecting plagiarized passages translated from English to Persian. Further improvements can also be done by integrating *Hamtajoo* and *Maglet* [21] that is a Persian journal recommender system. It can facilitate the manuscript submission process for academicians by checking plagiarism and also by finding appropriate journals for their manuscripts in an integrated system.

Acknowledgment. This work has been accomplished in ICT research institute, ACECR and funded by Vice Presidency for Science and Technology of Iran - Grant No. 1164331. We would like to thank all of the members of ITBM and AIS research groups of ICT research institute for their contribution in corpus construction. Especial credit goes to Javad Rafiei and Khadijeh Khoshnava for their help in development and testing the algorithms.

References

1. Brown, N., Janssen, R.: Preventing plagiarism and fostering academic integrity: a practical approach. J. Perspect. Appl. Acad. Pract. **5**(3), 102–109 (2017)
2. Lancaster, T., Culwin, F.: Classifications of plagiarism detection engines. Innov. Teach. Learn. Inf. Comput. Sci. **4**(2), 1–16 (2005)
3. Lukashenko, R., Graudina, V., Grundspenkis, J.: Computer-based plagiarism detection methods and tools: an overview, In: Proceedings of the 2007 International Conference on Computer Systems and Technologies, CompSysTech 2007, Rousse, Bulgaria, 14–15 June 2007
4. Zhang (Yuehong), H.: CrossCheck: an effective tool for detecting plagiarism. Learn. Publ. **23**(1), 9–14 (2010)
5. Zhang, Y., Jia, X.: A survey on the use of CrossCheck for detecting plagiarism in journal articles. Learn. Publ. **25**(4), 292–307 (2012)
6. Batane, T.: Turning to Turnitin to fight plagiarism among university students. Educ. Technol. Soc. **13**(2), 1–12 (2010)

7. Farghaly, A.: Computer processing of Arabic script-based languages: current state and future directions. In: Proceedings of the Workshop on Computational Approaches to Arabic Script-Based Languages (2004)
8. Menai, M.E.B., Bagais, M.: APlag: a plagiarism checker for Arabic texts. In: 6th International Conference on Computer Science and Education (ICCSE), pp. 1379–1383 (2011)
9. Alzahrani, S., Salim, N., Abraham , A., Palade, V.: iPlag: intelligent plagiarism reasoner in scientific publications. In: 2011 World Congress on Information and Communication Technologies (WICT), pp. 1–6 (2011)
10. Meuschke, N., Gipp, B., Breitinger, C., Berkeley, U.: CitePlag: a citation-based plagiarism detection system prototype. In: Proceedings of the 5th International Plagiarism Conference (2012)
11. Si, A., Leong, H.V., Lau, R.W.H.: CHECK: a document plagiarism detection system. In: Proceedings of the 1997 ACM Symposium on Applied Computing (SAC 1997), San Jose, CA, USA, pp. 70–77 (1997)
12. Pera, M.S., Ng, Y.K.: SimPaD: a word-similarity sentence-based plagiarism detection tool on Web documents. Web Intell. Agent Syst. **9**(1), 27–41 (2011)
13. Collberg, C.S., Kobourov, S.G., Louie, J., Slattery, T.: SPLAT: a system for self-plagiarism detection. In: Proceedings of the International Conference WWW (ICWI), Algarve, Portugal, pp. 508–514 (2003)
14. Stein, B., zu Eissen, S.M., Potthast, M.: Strategies for retrieving plagiarized documents. In: Proceedings of the 30th Annual International ACM Conference on Research and Development in Information Retrieval (SIGIR), Amsterdam, The Netherlands, pp. 825–826 (2007)
15. Potthast, M., et al.: Overview of the 4th international competition on plagiarism detection. In: Evaluation Labs and Workshop (CLEF), Rome, Italy (2012)
16. Mohtaj, S., Roshanfekr, B., Zafarian, A., Asghari, H.: Parsivar: a Language Processing toolkit for Persian. In: Proceedings of the 11th International Conference on Language Resources and Evaluation (LREC), Miyazaki, Japan (2018)
17. Matsuo, Y., Ishizuka, M.: Keyword extraction from a single document using word co-occurrence statistical information. Int. J. Artif. Intell. Tools **13**(1), 157–169 (2004)
18. Witten, I.H., Paynter, G.W., Frank, E., Gutwin, C., Nevill-Manning, C.G.: KEA: practical automatic keyphrase extraction. In: Proceedings of the Fourth ACM Conference on Digital Libraries, Berkeley, CA, USA, pp. 254–255 (1999)
19. Hagen, M., Potthast, M., Stein, B.: Source retrieval for plagiarism detection from large web corpora: recent approaches. In: Conference and Labs of the Evaluation forum (CLEF), Toulouse, France (2015)
20. Asghari, H., Mohtaj, S., Fatemi, O., Faili, H., Rosso, P., Potthast, M.: Algorithms and corpora for persian plagiarism detection: overview of PAN at FIRE 2016. In: Forum for Information Retrieval Evaluation, Kolkata, India (2016)
21. Mohtaj, S., Tavakkoli, F.: Maglet: a Persian journal recommender system. In: 9th International Symposium on Telecommunications (IST), pp. 348–352 (2018)

Analyzing Stylistic Variation Across Different Political Regimes

Liviu P. Dinu and Ana Sabina Uban(✉)

Faculty of Mathematics and Computer Science,
Human Language Technologies Research Center, University of Bucharest,
Bucharest, Romania
auban@fmi.unibuc.ro

Abstract. In this article we propose a stylistic analysis of texts written across two different periods, which differ not only temporally, but politically and culturally: communism and democracy in Romania. We aim to analyze the stylistic variation between texts written during these two periods, and determine at what levels the variation is more apparent (if any): at the stylistic level, at the topic level etc. We take a look at the stylistic profile of these texts comparatively, by performing clustering and classification experiments on the texts, using traditional authorship attribution methods and features. To confirm the stylistic variation is indeed an effect of the change in political and cultural environment, and not merely reflective of a natural change in the author's style with time, we look at various stylistic metrics over time and show that the change in style between the two periods is statistically significant. We also perform an analysis of the variation in topic between the two epochs, to compare with the variation at the style level. These analyses show that texts from the two periods can indeed be distinguished, both from the point of view of style and from that of semantic content (topic).

Keywords: Authorship attribution · Stylome classification · Stylistic variation · Romanian

1 Previous Work and Motivation

Automated authorship attribution has a long history (starting from the early 20th century [12]) and has since then been extensively studied and elaborated upon. The problem of authorship identification is based on the assumption that there are stylistic features that can help distinguish the real author of a text from any other theoretical author. This said set of stylistic features was recently defined as a linguistic fingerprint (or stylome), which can be measured, is largely unconscious, and is constant [18]. One of the oldest studies to propose an approach to this problem is on the issue of the *Federalist Papers*, in which the authors [13] try to determine the real author of a few of these papers which have disputed paternity. This work remains iconic in the field, both for introducing a standard dataset and for proposing an effective method for distinguishing between the authors' styles, that is still relevant to this day, based on function words frequencies. Many other types of features have been proposed and successfully used in subsequent studies to determine the author of a text. These types of features generally contrast with the content words commonly used in text categorization by topic, and are said to be used unconsciously and harder to control by the author. Such features are, for example, grammatical structures [1], part-of-speech n-grams [7], lexical richness [17], or even the more general feature of character n-grams [5,6]. Having applications that go beyond finding the real authors of controversial texts, ranging from plagiarism detection to forensics and security, stylometry has widened its scope into other related subtopics such as author verification (verifying whether a text was written by a certain author) [8], author profiling (e.g. gender or age prediction), author diarization, or author masking (given a document, paraphrase it so that the original style does not match that of its original author anymore).

In this work we attempt to explore a slightly different issue, strongly related with the stylistic fingerprint of an author, namely if an author preserves his stylome across different time periods, and across political and cultural environments. More specifically, we want to see if we can discriminate between texts written by the same author under a communist regime, as compared to under democracy.

For this analysis, we chose the works of Solomon Marcus (1925–2016), one of the most prominent scientists and men of culture of contemporary Romania, and a very prolific author whose work spans a large period of time, with a consistent amount of work both before and after the fall of communism in Romania. As a scientist, he was active in an impressive range of different fields, such as mathematics, computer science, mathematical and computational linguistics, semiotics etc., and published over 50 books translated in more than 10 languages, and about 400 research articles in specialized journals in almost all European countries. He is one of the initiators of mathematical linguistics [11] and of mathematical poetics [9], and has been a member of the editorial board of tens of international scientific journals covering all his domains of interest. As a man of culture, he has written an equally impressive amount of texts, having as preferred topics mathematical education, culture, science, children, etc. His wide interests and complex personality have left deep imprints in the Romanian scientific and cultural world.

We chose these works and this period in Romanian history as a particularly interesting study case for stylistic variation. Marcus' essays [10] were written over a period spanning almost half a century: 22 years of communist regime and 27 years of democracy (after the fall of communism in Romania in 1989). The significant changes in Romania's cultural life with the fall of communism motivate such an analysis beyond a simple temporal analysis of texts: in the pre-democracy period, the communist norms demanded an elaborated writing style, whereas the list of officially approved topics was much more limited. On the other hand, with the fall of communism, we expect that the author would be able to radically change his style to express his ideas freely.

Previous work on the same corpus [2] focused on a lexical quantitative analysis of Marcus' works, uncovering some differences in word usage between the communist and the democracy period. We continue this work by performing more robust analyses using various metrics and classification experiments to show that the texts written during the two periods are indeed distinguishable with a reasonable accuracy.

2 The Corpus

For the purposes of the analysis presented here, we used a collection of Solomon Marcus' non-scientific texts. The whole collection of Marcus' non-scientific publications is available in print, in six volumes entitled "Răni deschise" (*Open wounds*), of which we analyze the first four: two containing texts written before the fall of communism, and the other two with texts written under democracy. The first volume comprises 1256 pages of texts, conferences and interviews, from 2002 to 2011 (and a few published before 1990). The second volume "Cultură sub dictatură" (*Culture under dictatorship*) of 1088 pages and the third "Depun Mărturie" (*I testify*), of 668 pages, is a collection of texts published between 1967 and 1989. The fourth volume "Dezmeticindu-ne" (*Awaking*), of 1030 pages, contains texts from the period immediately following the Romanian Revolution (December 1989) to the year 2011.

Before the texts could be used for automatic analysis, some pre-processing was involved, including extracting the texts from PDF files into text form, and splitting the corpus into individual essays. We also parsed the table of contents in order to label each text with its year of publication. Additionally, we excluded some texts that would interfere with the experiments, such as texts in languages different than Romanian (most of the texts are in Romanian), and interviews (which also contain lines of the interviewer, that should not be considered when analysing text authored by Solomon Marcus).

3 Methodology

In order to analyze the style of Marcus' work and its variation, we perform a series of classification and clustering experiments, as well as a temporal analysis of the evolution of the author's style. We look at various features to capture both style and semantic content, with a focus on metrics that characterize the stylistic fingerprint of the author.

We perform various classification and clustering experiments, as well as visualisations complementing them, for which we experiment first with simple bag of words models. To capture the stylistic fingerprint of the author, we resort to one of the most commonly accepted indicators of personal style of an author, namely the distribution of functional words, such as pronouns, determiners, prepositions, etc., which were successfully used in many previous studies to solve authorship attribution problems. For comparison, we perform a parallel experiment where we use as features only the content words, assuming they are descriptive of the semantic content of the texts, as opposed to their style.

Aside from these feature sets, we also use more sophisticated features in some additional classification experiments: various stylistic metrics (such as readability or lexical richness) to capture the stylistic variation, as well as features resulted from topic modelling to capture the semantic content. These features will be described in more detail in the chapters dedicated specifically to the classification experiments.

3.1 Clustering Experiments

Processing the text involves extracting all function words from each essay, and representing each text as a vector of frequencies of function words. We use a curated list of 120 functional Romanian words [4]. On these vectors we then apply similarity metrics in order to cluster texts based on their stylistic profile. The distance we have chosen is rank distance [3,14,15], an ordinal distance which was successfully previously used in other problems of authorship [4,5].

For our purposes, we applied a hierarchical clustering algorithm, and plotted the resulted dendrogram, as shown in Fig. 1. Texts from each volume are marked with the prefix $RD < volume\_number >$. The intention is to see whether texts from the same period are clustered together: in this case, we would expect to see texts from volumes 1 and 4 (communism) appear close together, and texts from volumes 2 and 3 (post-communism) appear in a separate group.

An additional processing step we performed before applying the clustering algorithm was to group the essays into longer texts, while keeping them separated into volumes (texts from different volumes/periods were not grouped together). For each volume, batches of 10 essays were grouped together in 1 text. This results in a fewer number of texts, which can be more easily plotted on a dendrogram, and additionally should contain less noise than the very short original texts.

3.2 Dimensionality Reduction and Visualization

In addition to the clustering experiments, we perform an experiment where we plot the essays in the 4 volumes in 2-D space, by initially applying dimensionality reduction (principal component analysis) on the texts, represented as function word frequency vectors, as described above. The resulted plot is shown in Fig. 2.

In a separate experiment, we take a brief look at the content profile of the texts (as opposed to the stylistic one), by performing the same analysis on content words instead of function words. We extract all content words from all the (grouped) texts, and represent the texts as vectors of content words frequencies. We then apply PCA on the resulted vector space, and plot the 2-dimensional representations of the texts, as shown in Fig. 3. Although this is still a very crude analysis, the result should provide an insight into the content similarities and dissimilarities between the texts in the 4 volumes.

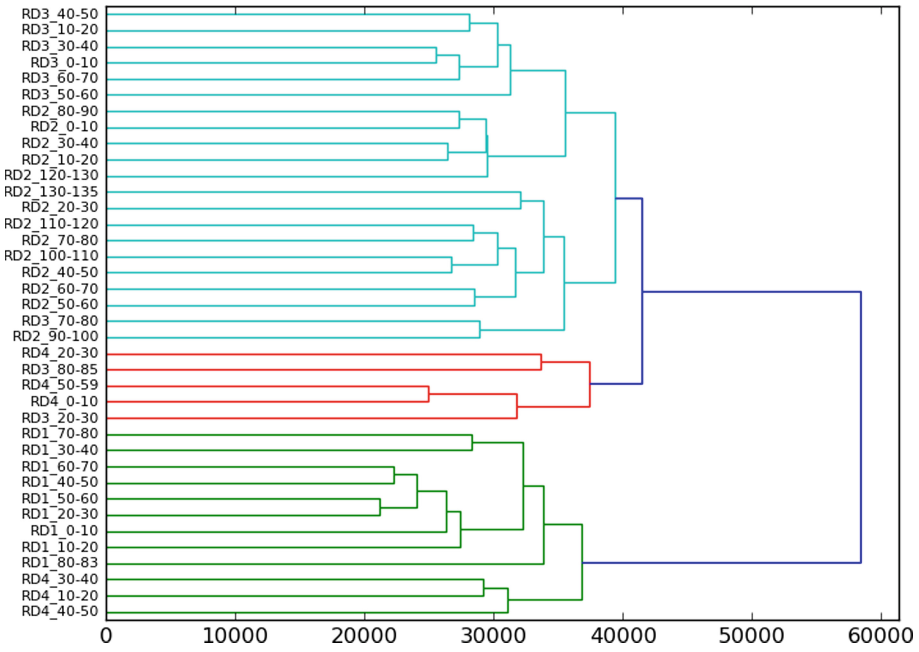


Fig. 1. Style dendrogram: hierarchical clustering of function word vectors

We also perform a version of the clustering experiments on the same vectors of content words, with the same methodology that was applied in the stylistic analysis: the rank distance similarity is applied on the word vectors, and the distances are used in a hierarchical clustering algorithm, which results in the "content" dendrogram in Fig. 4.

3.3 Classification Experiments

In order to have a more conclusive result regarding the differences between the texts written in the two epochs and the levels at which they occur, we also performed some classification experiments where we try to predict for each essay if it was written before or after the fall of communism, as well as to which specific volume it pertains.

For classifying the texts we used an SVM classifier with a linear kernel, and tested its performance in a series of leave-one-out experiments.

3.3.1 Features

We used various sets of features, in turn specific to style and to semantic content, in order to conclusively characterize the nature of the evolution of Marcus' writing over time and across the two periods.

As stylistic features, we experimented with 2 setups: first a bag-of-words model using only functional words; and secondly a set of stylistic markers:

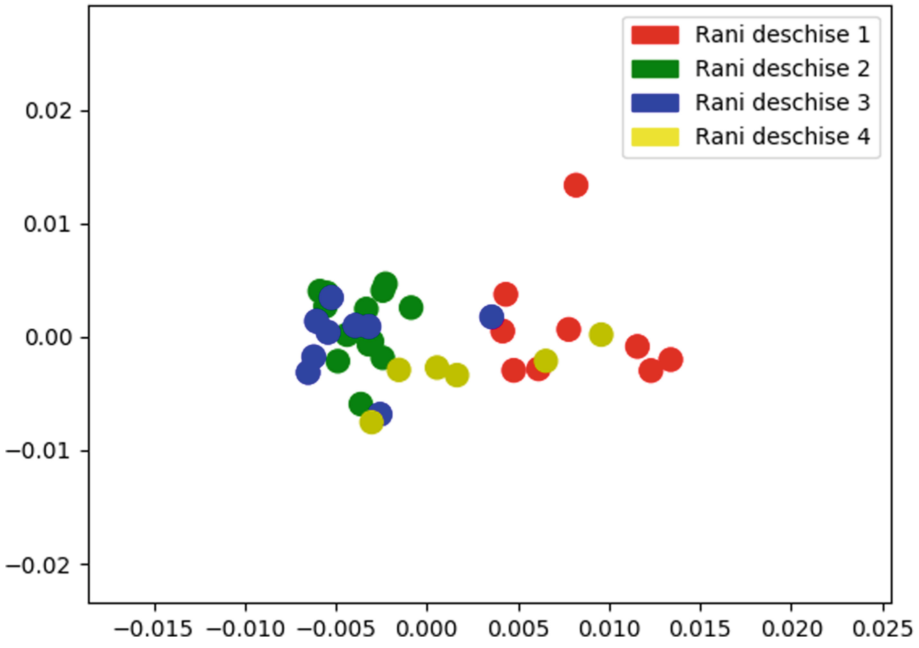


Fig. 2. Style space: 2-D visualization of function word vectors

– **Automated Readability Index**, as defined in [16], computed as:

$$ARI = 4.71\frac{c}{w} + 0.5\frac{w}{s} - 21.43$$

where c, w and s represent, respectively, total number of characters, words and sentences in the text

– **Lexical richness**, computed as the number of unique lemmas divided by the total number of tokens in the text:

$$LR = \frac{number\ of\ unique\ lemmas}{total\ number\ of\ tokens}$$

– **Average word length**
– **Average sentence length**

To measure the variation in semantic content, we performed two classification experiments. In the first experiment we used a simple bag-of-words model, this time using only content words as features. For the second experiment, we used topic modelling to extract topics in each article, that we then used as features in the classifier. To extract the topics, we used a Latent Dirichlet Allocation model, configured to generate 3 latent topics for each document.

Results are shown in Table 1 and Table 2: in Table 1 we report the results of trying to classify the texts into the two distinct periods, while Table 2 shows the results of classifying each text into the specific volume to which it pertains.

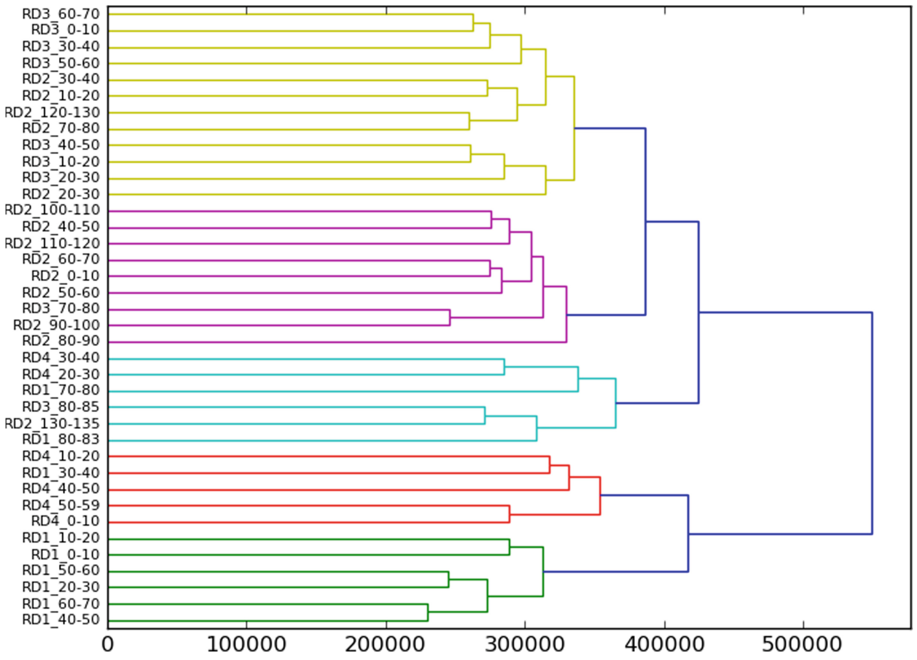


Fig. 3. Topic dendrogram: hierarchical clustering on content word vectors

3.4 Temporal Variation of Stylistic Metrics

In order to show that the stylistic variation discovered in the experiments described in the previous chapter can indeed be attributed to the change of political regime, and not simply to the passing of time and the natural gradual change in the author's style, we also look at how the various stylistic metrics vary over time at a finer level.

Figures 5, 6, 7 and 8 show the evolution of each stylistic metric described in the previous chapter from year to year. To achieve this, we computed each metric for every article in the corpus, then for each year we averaged the values obtained for the articles written during that year. To reduce the effect of noise and variation in number of articles for each year, we smooth the obtained values by applying a moving average over the articles in each year with a window of 10 years.

In order to confirm the variation in style is more pronounced around the year 1990 (that marks the beginning of democracy in Romania), we performed a series of experiments in which we attempted temporal splits of the article set into two periods using each year as a separator, and computed the statistical significance of the difference shown at the level of each metric for each of these splits. We assume that the more significant the split across a certain year is, the more radical the author's change in style around that year, and compare the effect of the particular year that marked the beginning of democracy in

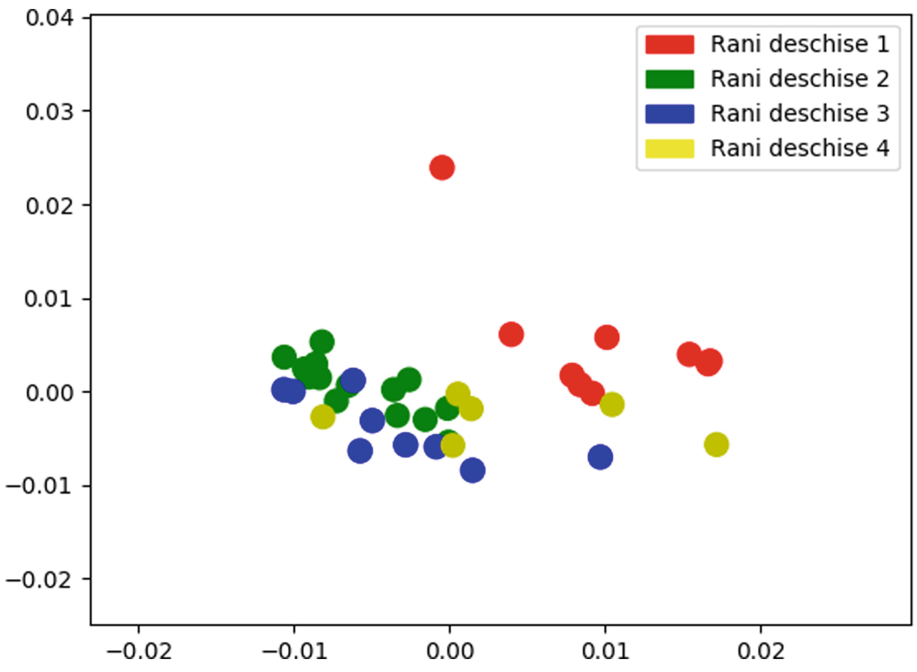


Fig. 4. Topic space: 2-D visualization of content word vectors

Romania to all of the other years during which Marcus wrote his essays. We plot, in addition to the actual values of the stylistic metrics for each year, the p-values corresponding to the statistical significance of the difference for each metric between the articles written before and after each year. The vertical line over the year 1990 splits the plot into two parts, corresponding to the period before and after the fall of communism, respectively.

4 Results and Analysis

The clustering experiments, as well as the 2-dimensional visualizations, show a clear distinction between the works of the two periods. The style dendrogram in Fig. 1 shows that texts tend to be grouped according to the volume they belong to, but moreover, the volumes belonging to each period tend to be grouped together. It is noticeable that texts belonging to the first and fourth volumes, written during the post-communist epoch (marked "RD1" and "RD4"), are grouped together on the lower branch of the dendrogram, and the same happens with texts in volumes 2 and 3 ("RD2" and "RD3"), written during communism, which are grouped on the upper branch.

The same tendency is visible in the 2-dimensional vector space plot of the texts. It is interesting that the separation between the pre-communist texts (marked with green and blue points) and communist texts (marked with red

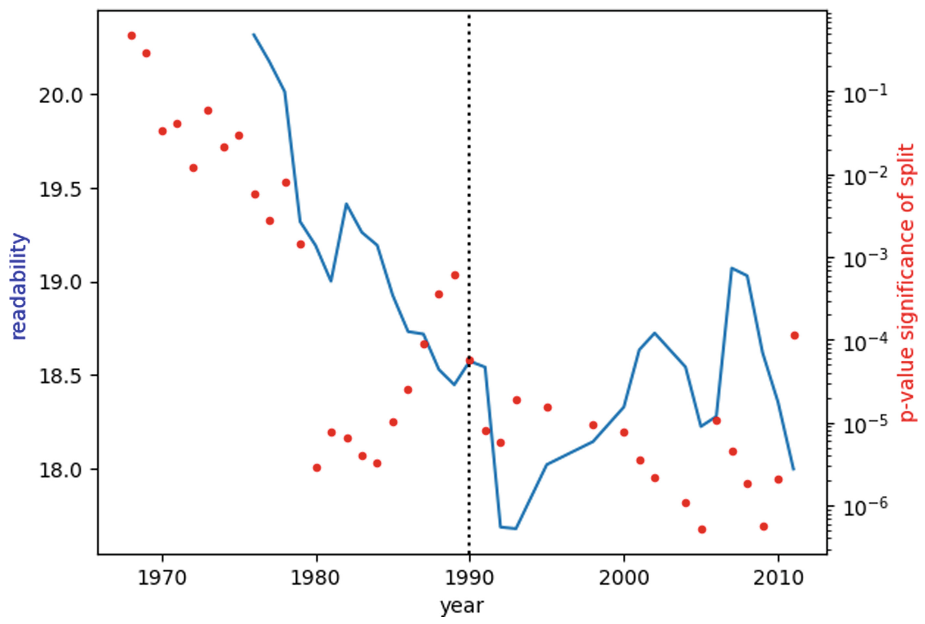


Fig. 5. Evolution of readability over time

and yellow points) seems even more prominent than the separation between each of the four volumes - indicating that the main variable contributing to style differences is indeed the period when the text was written.

A better, more detailed understanding of how exactly Marcus' style evolves can be gained from the plots showing the variation in style from year to year. Overall, after the change of regime in 1990, we can see a slight drop in all of the stylistic features, especially in word length and lexical richness. Most of the stylistic metrics (especially lexical richness and readability) show a more pronounced change of style right after the year 1990 - the year of the fall of the communist regime in Romania - suggesting that the change of regime did have an effect of its own on the style of the author. An even more interesting phenomenon can be observed by looking at the variation of the p-values for splitting the article set into two arbitrary periods (before and after each year). Except from average sentence length, for which the most drastic change seems to happen around the year 1980, for all of the other stylistic metrics splitting the articles set into texts written under communism and under democracy is statistically significant (with p-values well under 0.05 for the year 1990, and additionally having at least local minima around this year).

We can conclude from these results that, irrespective of possible variation at other levels (topic, etc.), there is inarguable variation between the texts at a stylistic level.

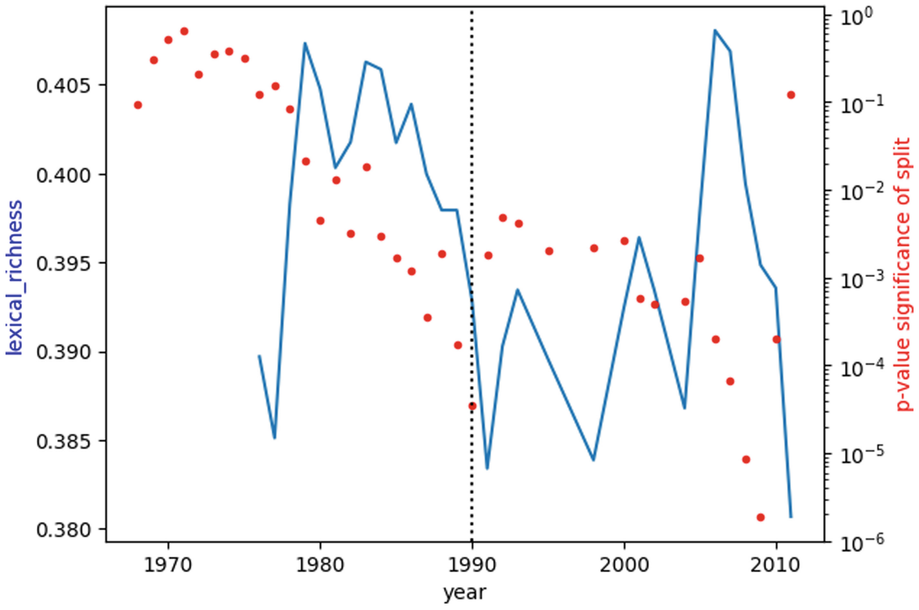


Fig. 6. Evolution of lexical richness over time

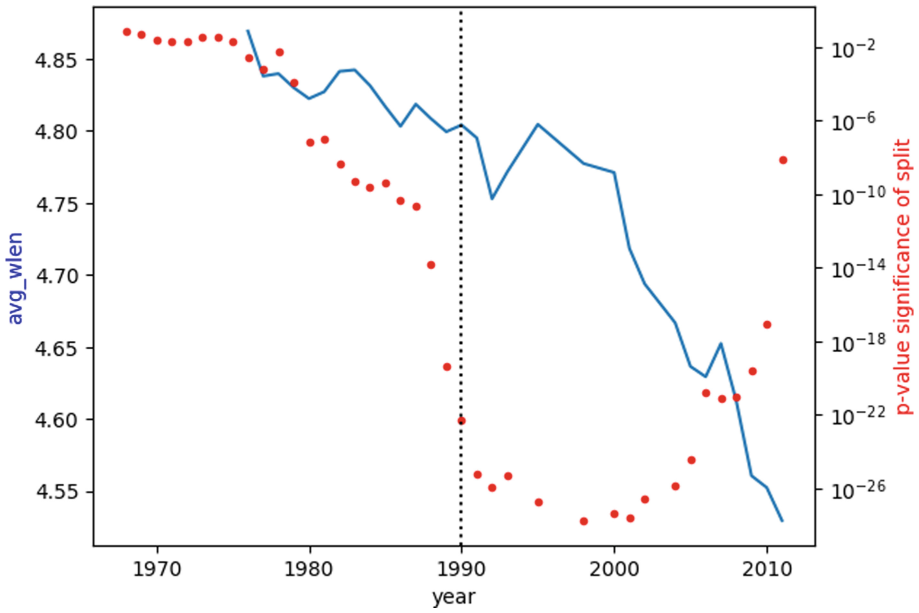


Fig. 7. Evolution of average word length over time

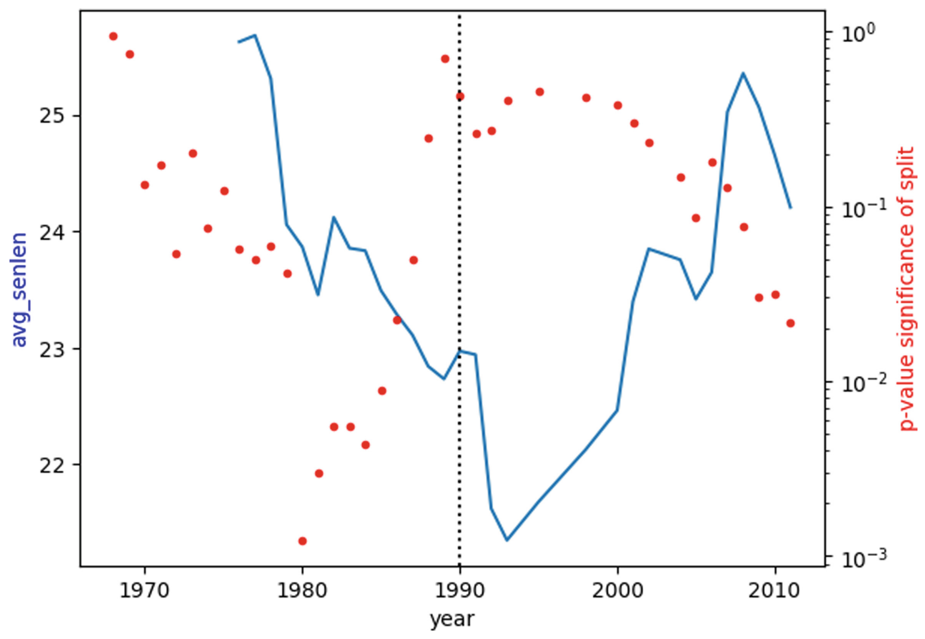


Fig. 8. Evolution of average sentence length over time

A similar but less obvious effect is noticeable in the content analysis. While the vector space plot of the texts again shows a reasonably clear distinction between the texts in the two periods, the cluster analysis is less conclusive.

Table 1. Results of classifying texts into periods

Feature set	Precision
Function words (style)	75.80%
Stylistic metrics (style)	73.12%
Content words (semantic content)	74.46%
Topic modelling (semantic content)	71.24%

The results of the classification experiments very clearly show that essays can be easily separated into the two classes corresponding to communism and democracy, both when using function words, stylistic metrics, and semantic features (content words, topics) as a feature set. A slightly better accuracy is obtained for stylistic classification. The stylistic features, in comparison to the content words, seem to be more successful at predicting the period rather than the specific volume, indicating that style may indeed be the aspect of the texts that varies the most between the two periods and political regimes. Analyses of the

Table 2. Results of classifying texts into volumes

Feature set	Precision
Function words	61.29%
Content words	84.13%

author's change in style, as well as statistical significance tests of the change in various stylistic metrics across the two periods, further confirm that the change of political regime does have its own effect on the author's style.

We can conclude, given all of these results, that during the two periods, corresponding to two different political regimes, Marcus produced works which are clearly distinct both stylistically and from the point of view of the topics discussed. It is noteworthy that the classifier manages not only to classify the texts into the two periods in which they were written, but is also successful at predicting the exact volume to which each essay pertains. This suggests there may be other (volume-specific) factors contributing to the distinctiveness of the essays in each class, which it may be interesting to further explore.

5 Conclusions and Future Directions

We have proposed in this paper an analysis of the stylistic variation of the works of Solomon Marcus between two distinct historical periods, and have presented results that show the separation between the two periods is indeed clear at the level of the author's style. The clustering approach based on the preference of the author regarding the functional words shows that the functional words were unconsciously used by the author in a different way in communism than in democracy period, and we can use them to discriminate between the communist and post-communist texts of the given author. We further confirm in several classification experiments that the texts can be automatically separated into the two periods, and are distinguishable both at the level of style and topic. We show the contribution of the change in political regime to the stylistic variation by looking at how the author's style gradually changes over time, and by performing statistical significance tests on various stylistic metrics, comparing the two periods.

In future work, it would be very interesting to examine more closely how this variation is manifested. From the stylistic point of view: what are the exact changes that occur with the passing of time and the change of historical and cultural context? From the point of view of topic modelling, what are the specific changes in the topic of the text corresponding to the two periods? Do these changes match our intuitions about writing in the communist as compared to post-communist era?

Finally, continuing this analysis on other authors with works in both communism and post-communism, or, more generally, across various political regimes, would be interesting to confirm the phenomenon is not unique to this author.

Possibly extending this to other authors in different cultural contexts could lead to an interesting analysis on the external factors that influence style.

Acknowledgement. Liviu P. Dinu is supported by UEFISCDI, project number 53BG/2016.

References

1. Baayen, H., Van Halteren, H., Tweedie, F.: Outside the cave of shadows: using syntactic annotation to enhance authorship attribution. Literary Linguist. Comput. **11**(3), 121–132 (1996)
2. Dinu, A., Dinu, L.P., Dumitru, B.: On the stylistic evolution from communism to democracy: Solomon Marcus study case. In: Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017, pp. 201–207. INCOMA Ltd., Varna, Bulgaria, September 2017. https://doi.org/10.26615/978-954-452-049-6_028
3. Dinu, L.P.: On the classification and aggregation of hierarchies with diifferent constitutive elements. Fund. Inform. **55**(1), 39–50 (2003)
4. Dinu, L.P., Niculae, V., Şulea, O.M.: Pastiche detection based on stopword rankings: exposing impersonators of a Romanian writer. In: Proceedings of the Workshop on Computational Approaches to Deception Detection, pp. 72–77. Association for Computational Linguistics (2012)
5. Dinu, L.P., Popescu, M., Dinu, A.: Authorship identification of Romanian texts with controversial paternity. In: LREC (2008)
6. Kešelj, V., Peng, F., Cercone, N., Thomas, C.: N-gram-based author profiles for authorship attribution. In: Proceedings of the Conference Pacific Association for Computational Linguistics, PACLING, vol. 3, pp. 255–264 (2003)
7. Koppel, M., Schler, J.: Exploiting stylistic idiosyncrasies for authorship attribution. In: Proceedings of IJCAI'03 Workshop on Computational Approaches to Style Analysis and Synthesis, vol. 69, p. 72 (2003)
8. Koppel, M., Schler, J.: Authorship verification as a one-class classification problem. In: Proceedings of the Twenty-First International Conference on Machine Learning, p. 62. ACM (2004)
9. Marcus, S.: Mathematische poetik, vol. 151977. Editura Academiei (1973)
10. Marcus, S.: Rani deschise. Editura Spandugino (2012–2017)
11. Marcus, S., Nicolau, E., Stati, S.: Introduzione alla linguistica matematica. Casa editrice R. Pàtron, Bologna (1971)
12. Mendenhall, T.C.: A mechanical solution of a literary problem. Popular Sci. Mon. (1901)
13. Mosteller, F., Wallace, D.L.: Inference in an authorship problem: a comparative study of discrimination methods applied to the authorship of the disputed federalist papers. J. Am. Stat. Assoc. **58**(302), 275–309 (1963)
14. Popescu, M., Dinu, L.P.: Kernel methods and string kernels for authorship identification: the federalist papers case. In: Recent Advances in Natural Language Processing, p. 484 (2007)
15. Popescu, M., Dinu, L.P.: Rank distance as a stylistic similarity. COLING (Posters) **8**, 91–94 (2008)
16. Senter, R., Smith, E.A.: Automated readability index. Technical report, CINCINNATI UNIV OH (1967)

17. Tweedie, F.J., Baayen, R.H.: How variable may a constant be? Measures of lexical richness in perspective. Comput. Humanit. **32**(5), 323–352 (1998)
18. Van Halteren, H., Baayen, H., Tweedie, F., Haverkort, M., Neijt, A.: New machine learning methods demonstrate the existence of a human stylome. J. Quant. Linguist. **12**(1), 65–77 (2005)

Towards Automated *Fiqh* School Authorship Attribution

Maha Al-Yahya(✉)

Information Technology Department, College of Computer and Information Sciences, King Saud University, Riyadh, Saudi Arabia
malyahya@ksu.edu.sa

Abstract. The word *Fiqh* (Islamic jurisprudence) refers to the body of Islamic law (Shari'ah). A large volume of Fiqh literature has been generated over the past thirteen hundred years, some of which texts have unknown authors. The importance of identifying the Fiqh School emanates from its importance in offering an authenticated interpretation of fundamental sources.

The traditional method for identifying the Fiqh School for a certain text is either by knowledge of the school affiliation the author or by close reading of the text by Fiqh scholars. This method is costly in terms of the time and human effort involved. An alternative to this manual approach is automated identification of Fiqh school texts using stylometric analysis. In this study we investigate the extent to which stylometric features can be used as predictors for Fiqh school authorship of a given text. We explore a corpus of Arabic Fiqh texts using unsupervised cluster analysis and supervised machine learning.

The results of our study show that the Fiqh schools have distinctive text style features that can be used to indicate authorship. The observations from the cluster analysis experiment using a number of different distance measures are visualized using network graphs. The best clustering in terms of Fiqh school division was achieved by the Classic Delta distance measure and Eder's Delta distance measure. The results from the supervised experiment comparing the four classification algorithms: Support Vector Machines (SVM), Naïve Bayes (NB), K-Nearest Neighbor (KNN), and Delta show that supervised classification using SVM produces the highest average accuracy at 97.5% for the task of Fiqh school prediction.

Keywords: Fiqh school attribution · Stylometric Analysis · Arabic language · Cluster analysis · Supervised Classification

1 Introduction

The word Fiqh (Islamic jurisprudence) refers to the body of Islamic law (Shari'ah) that is derived from the two fundamental sources of Islam, the Qur'an (the divine word of God) and the Hadith (the authenticated prophetic teachings and practices). In the context of Islamic law (*Shari'ah*) studies, the importance of identifying the Fiqh School emanates from the importance of Sharia law in the everyday lives of Muslims. There exists a large volume of Fiqh literature generated over thirteen hundred years, for some of which the author is unknown, and thus the Fiqh School is also unknown. To be able to utilize these texts in jurisprudence studies, it is vital to be able to recognize whether a given text is of the Fiqh School. The traditional method for identifying the Fiqh School for a certain text is either by knowledge of the text author school or by close reading of the text by Fiqh scholars. This method is costly in terms of the time and human effort involved. An alternative to this manual approach is to use automated identification of Fiqh school texts using stylometric analysis.

In this study we investigate the extent to which stylometric features can be used as a predictor for Fiqh school authorship of a given text. We explore a corpus of Arabic Fiqh texts from the four Sunni Fiqh schools (the Maliki, the Hanafi, the Shafi'i, and the Hanbali) using cluster analysis and supervised machine learning techniques. The stylometric features used are the most frequent words in the corpus (MFWs) and the cluster analysis is applied using a number of distance measures. For the supervised experiment, four machine learning algorithms are applied: Support Vector Machines (SVM), Naïve Bayes (NB), K-Nearest Neighbor (KNN), and Delta Classifier (specifically designed for the humanities literature [1]), are applied and compared.

The remainder of this article is organized as follows: Sect. 2 presents background and related work in the area of stylometric analysis. Section 3 presents the methodology and experiment. Section 4 presents the results and evaluation and finally, Sect. 5 presents conclusions and future work.

2 Related Work

Stylometric analysis is a form of computational modeling, more precisely it is a “quantitative approach to textual corpora” [2]. It involves “the extraction of the most appropriate features which can provide quantitative information about an author's style” [3]. Stylometry, the analysis of authorship style [4], is based on the assumption that it is possible to identify an author's style by studying the language the author uses – the stylometric features [5]. Stylometric analysis originated in the field of authorship attribution; however, it has been applied to other fields including textual forensics [6, 7], literary influence [8], genre detection [8], political affiliation [9], and comparative literature and translation studies [2].

In this study we leverage stylometric analysis to explore the Fiqh schools writing style in the field of Fiqh studies and investigate how quantitative methods can be applied in Islamic jurisprudence research. The problem of Fiqh school attribution can be viewed as an “author profiling” problem. Author profiling refers to a characteristic or property of the author, as Juola [10] points out “determining any of the properties of the author(s) of a sample of text”. Authorship profiling can be defined as “the analysis of a document to determine certain demographics such as gender without directly identifying the author” [4].

There are limited studies in the area of author profiling for Arabic. The work presented in [9] describes text classification for the task of political orientation determination. The authors study the problem of political orientation classification from both a text classification approach and a stylometric approach, and compare the results. The dataset they used includes articles and posts are collected from social networks and forums.

For the stylometric approach, the authors select a set of features including lexical features (characters and words), syntactic features, structural features, and content-specific words. They conclude that the text classification approach is superior, with an accuracy of 83.9%, compared to the stylometric approach, with an accuracy of 60%. They suggest that different kinds of feature reductions, such as stemming and feature selection, have negative effects on the results.

Another study on ideological and organizational affiliation classification for Arabic text is presented in [11]. The authors use a stylometric approach to address the problem. The dataset is composed of 2 corpora, one for organizational affiliation with documents categorized into 4 categories, and one for ideological streams categorized into four different categories, both manually labeled. The features selected were the set of 1000 most frequent words (MFWs) of the corpus. No stemming has been applied. The study employed the Bayesian multi-class regression (BMR) supervised learning method. For organization identification, the accuracy is 100% with 1000 MFW, and with 82 MFW the accuracy is 80%. The results indicate that stylometric features represent a robust and reliable method for categorization of Arabic text into organizational or ideological streams. However, the authors did not apply the classification to other classification models.

The study presented in [12] describes an experiment on author profiling for Arabic emails. The study presents machine learning classifiers for different author traits including age, gender, education, and psychometric traits. Comparing the classifiers, the best performance is achieved by SVM and Bagging. Another study which focuses on author gender identification for Arabic is presented in [13]. The authors view the problem as a classification problem and a number of classification algorithms are applied on an Arabic dataset using the bag-of-words approach for feature extraction. The results show high accuracy for the SVM classifier in regard to the gender identification task.

3 Method

The process of stylometric analysis for Fiqh attribution is shown in Fig. 1. Using the Fiqh dataset, the text is first pre-processed. Pre-processing involves tasks such as tokenization, stemming, part of speech (POS), removing non-alphabetic characters and spaces, and converting uppercase letters to lowercase. For our study, the only pre-processing applied is tokenization and removing of diacritics as this study is intended to be a baseline for future studies in which other forms of pre-processing can be applied to the corpus to determine the impact of such on performance.

The next task of the stylometric analysis is feature extraction. Features are extracted from the pre-processed texts. For the experiment in this study, function words are used as features, which are the MFWs in the dataset, which have been reported as a baseline in the past and shown to yield good results in stylometric studies [10]. After feature selection, the analysis is performed and finally the results are presented. In the case of clustering, a clustering algorithm is selected and the clustering is applied on the complete dataset. The results are presented in a diagram, and in our study we represent the results as a network graph. In the case of classification, the corpus is divided into two sets; a training set to train the classifier and a test set for testing the accuracy of the Fiqh school

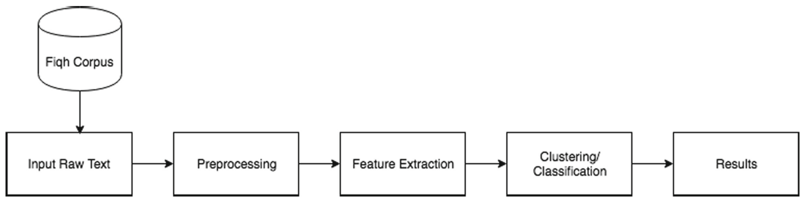


Fig. 1. Stylometric analysis process

prediction. The results of the classification are represented in the form of the accuracy of prediction.

The package Stylo [1], developed for the R environment for statistical computing [14], is used for the pre-processing, the extraction of the features, and the application of the clustering and classification analyses to the Fiqh dataset.

3.1 Dataset

The dataset for the study was extracted from *Almaktabah Al Shamela* website [15], which is an online digital library of Arabic texts from the early pre-Islamic era to the modern period covering a variety of subjects including language, religion, and science. A Fiqh corpus was developed by extracting all Fiqh texts available in the library[1], organized into the four major Sunni Fiqh schools -Hanafi, Maliki, Shafi'i, and Hanbali. The documents were converted into raw text format (txt) with no diacritics. No form of pre-processing or stemming was performed on the dataset except tokenization. The dataset includes a total of 135 books from the four Sunni Fiqh schools Hanbali, Hanafi, Shafi'i, and Maliki, the representation of each school in the corpus is 29%, 26%, 26%, and 19% respectively.

3.2 Cluster Analysis

Clustering is the process of dividing data into meaningful groups (clusters). In clustering analysis, a measure of nearness is computed between documents [16]. The choice of distance measure is an important step in clustering as it defines how the similarity between the two texts is computed and it will influence the shape of the clusters. There are a number of distance measures used in clustering for stylometric analysis which include: (1) Classic Delta (originally the Burrows' delta), (2) Eder's Delta, (3) Eder's Simple Delta, (4) Argamon's Delta, (5) Canberra distance, (6) Manhattan distance, (7) Euclidean distance, and the (8) Cosine distance. The cluster analysis method used is the bootstrap consensus tree (BCT) approach which is implemented in the Stylo package using 100–1000 MFWs as features. In this study cluster analysis is applied using the eight distance measures, and the results are compared to identify the best clustering among the four Fiqh schools.

Comparing the resulting network graphs, the best clustering in terms of Fiqh school division was achieved by the Classic Delta distance measure and Eder's Delta distance measure, as shown in Fig. 2, where the color denotes the actual Fiqh School (green for

[1] As of November 2017.

Hanbali, orange for Hanafi, purple for Shafi'i, and blue for Maliki). From the graph we can see that the texts are generally grouped into four clusters. Two of the clusters show a clear grouping of the Hanafi (orange) and Shafi'i (purple) Fiqh schools. For the other two clusters, one contains mixed texts from all four schools (central cluster), and the other shows two sub-clusters of Hanbali (green) and Maliki (blue) texts intermixed in one large cluster. The observations resulting from this initial cluster analysis experiment show that the Fiqh schools have distinctive style features that can be indicative of Fiqh school affiliation of the author.

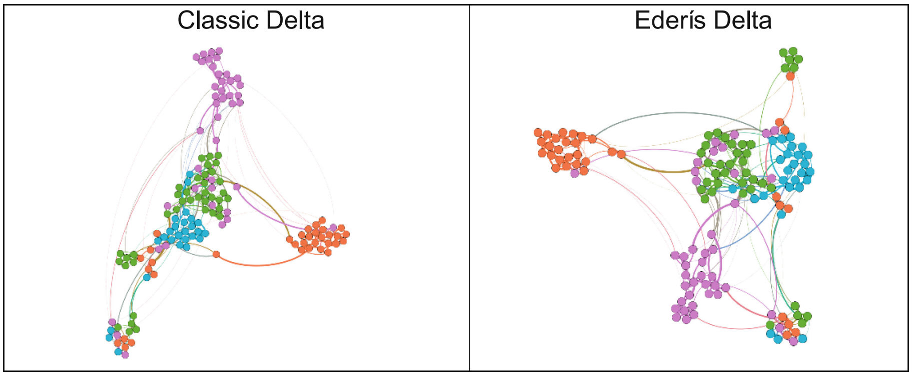


Fig. 2. Fiqh school clustering using classic delta and eder's delta

3.3 Classification

For classification, the dataset is divided into a training set (80%) and a test set (20%). Four classification algorithms are applied on the training set, and the resulting classifier is used to predict Fiqh school authorship in the test set. The predication accuracy is measured as the percentage of correct Fiqh school predictions. Stylo has a number of implementations for supervised machine learning classification algorithms which include Support Vector Machines (SVM), Naïve Bayes (NB), K-Nearest Neighbor (KNN), and Delta. A classifier was trained on the training dataset using each of these algorithms. For reliability, and in order to obtain more generalizable results, a 20-fold cross-validation was performed using the 100–1000 MFWs as features for the classifiers, with increments of 100 MFWs. The average accuracy of all classifiers is computed. Table 1 shows the results.

4 Evaluation

Results of the cluster analysis experiment show that a text which belongs to a certain Fiqh school has style features that can be discriminative of its Fiqh school authorship. The visualization of the network graphs generated from the clustering experiment show that the Classica Delta and Eder's Delta distance measures were able to generate the

Table 1. Fiqh school classification performance

	SVM	kNN, k = 1	kNN, k = 3	kNN, k = 5	kNN, k = 7	kNN, k = 9	NB	Delta
Average accuracy	97.4%	81.1%,	76.3%	61.9%	63.3%	65.2%	81.1%	93.7%

best group divisions based on text style. Eder's Delta has been designed to work with highly inflected languages such as Arabic [1]. Moreover, the intermixed cluster showing the closeness of the Hanbali and Maliki texts might be due to the assumption that the methodology used for reasoning in the interpretation of Islamic resources is similar in these two schools.

Regarding the classification experiment, results show that the best accuracy was achieved by the SVM classifier (97.4%) when using the set of 100–1000 words as features. The Delta classifier achieved very good results, giving an average accuracy of 93.7%. The best results for the kNN is achieved for k = 1 (81%). SVM is known to perform well and produces high accuracy for high-dimensional problems such as text classification [17]. The Delta classifier produced good results for the task of Fiqh school prediction, which might be due to the fact that this classifier has been tailored for use in humanities literature, of which the Fiqh corpus can be considered part.

5 Conclusion and Future Work

The work presented in this paper is a contribution to the quantitative methods that can be deployed in Islamic jurisprudence research. The aim of the work is to evaluate the existence of stylistic features of Fiqh texts that can support the task of Fiqh school attribution. Two experiments have been performed on a dataset of 135 Fiqh texts belonging to four different Fiqh schools. The clustering experiment showed that the style of a text is indicative of Fiqh School, and the best clustering is achieved using the Classical Delta and Eder's Delta. For the classification experiment, the best accuracy of prediction of the classifier was achieved by the SVM classifier, giving a prediction accuracy of 97.4%. The results also indicate that the use of the 100–1000 MFWs as features is suitable for the task of Fiqh school prediction.

One of the aims of this work is to develop a baseline for further development on Fiqh School attribution studies. For example, the only pre-processing implemented in this study was tokenization and diacritics removal; however, other pre-processing tasks can be performed which include stemming, tagging with part of speech, morphological analysis, and named entity extraction. The effect of pre-processing on classification performance can be analyzed. Moreover, the dataset can be extended to include more Fiqh texts.

References

1. Mike, K., Rybicki, J., Eder, M.: Stylometry with R: a package for computational text analysis. The R Journal. **8**, 107–121 (2016)

2. Wrisley, D.J.: Modeling the transmission of al-Mubashshir Ibn Fātik's Mukhtār al-Ḥikam in medieval europe: some initial data-driven explorations. J. Religion, Media and Digital Culture **5**, 228–257 (2016)
3. Stamatatos, E.: A survey of modern authorship attribution methods. J. Am. Soc. Inf. Sci. **60**, 538–556 (2009)
4. Neal, T., Sundararajan, K., Fatima, A., Yan, Y., Xiang, Y., Woodard, D.: Surveying stylometry techniques and applications. ACM Comput. Surv. **50**, 86:1–86:36 (2017)
5. López-Escobedo, F., Solorzano-Soto, J., Martínez, G.S.: Analysis of intertextual distances using multidimensional scaling in the context of authorship attribution. J. Quantitative Linguistics **23**, 154–176 (2016)
6. Rocha, A., et al.: Authorship attribution for social media forensics. IEEE Trans. Inf. Forensics Secur. **12**, 5–33 (2017)
7. Afroz, S., Brennan, M., Greenstadt, R.: Detecting Hoaxes, Frauds, and Deception in Writing Style Online. Presented at the May (2012)
8. Jockers, M.L.: Macroanalysis: Digital Methods and Literary History. University of Illinois Press, Urbana (2013)
9. Abooraig, R., Alwajeeh, A., Al-Ayyoub, M., Hmeidi, I.: On the Automatic Categorization of Arabic Articles Based on Their Political Orientation. Presented at the September 22 (2014)
10. Juola, P.: Authorship attribution. Found. Trends Inf. Retr. **1**, 233–334 (2006)
11. Koppel, M., Akiva, N., Alshech, E., Bar, K.: Automatically classifying documents by ideological and organizational affiliation. In: Proceedings of the 2009 IEEE International Conference on Intelligence and Security Informatics. pp. 176–178. IEEE Press, Piscataway, NJ, USA (2009)
12. Estival, D., Gaustad, T., Hutchinson, B., Pham, S.B., Radford, W.: TAT: an author profiling tool with application to Arabic emails. In: Proceedings of the Australasian Language Technology Workshop 2007., Melbourne, Australia (2007)
13. Alsmearat, K., Al-Ayyoub, M., Al-Shalabi, R.: An Extensive Study of the Bag-of-Words Approach for Gender Identification of Arabic Articles. Presented at the November (2014)
14. R Core Team: R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria (2016)
15. Almaktabah Alshamela, http://shamela.ws/. Accessed Jan 2018
16. Data Analysis and Data Mining: An Introduction. Oxford University Press, Oxford, New York (2012)
17. Joachims, T.: Text categorization with support vector machines: learning with many relevant features. In: Proceedings of the 10th European Conference on Machine Learning. pp. 137–142. Springer-Verlag, Berlin, Heidelberg (1998). https://doi.org/10.1007/BFb0026683

Stance and Gender Detection in Spanish Tweets

José-Ángel González, Lluís-F. Hurtado(✉), and Ferran Pla

VRAIN: Valencian Research Institute for Artificial Intelligence, Universitat Politècnica de València Camí de Vera s/n, 46022 València, Spain
{jogonba2,lhurtado,fpla}@dsic.upv.es

Abstract. In this paper, we present a deep learning based system for the user profiling and stance detection tasks in Twitter. Stance detection consists in automatically determining from text whether the author is in favor of a given target, against this target, or whether neither inference is likely. The proposed system assembles Convolutional Neural Networks and Long Short-Term Memory neural networks. We use this system to address, with minor changes, both problems. We explore embeddings and one-hot vectors at character level to select the best tweet representation.

We test our approach in the Stance and Gender Detection in Tweets on Catalan Independence track proposed at IberEval 2017 workshop. With the proposed approach, we achieve state-of-the-art results for the Stance detection subtask and the best results published until now for the Gender detection subtask.

Keywords: Stance detection · Gender detection · Deep learning · Twitter

1 Introduction

In recent years, Sentiment Analysis and Opinion Mining have aroused great interest in the Natural Language Processing community. Sentiment Analysis consists of determining the polarity (positive, negative or none) expressed in a text. In addition, the use of social networks to express opinions on any topic has also become widespread. Both facts have facilitated that workshops on Sentiment Analysis on Twitter have attracted the interest of a large amount of research groups.

SemEval workshop has devoted several tasks to the analysis of sentiments on Twitter at different levels in which the English and Arabic languages have been involved [18,19]. Also, for the Spanish language, within the conference of the SEPLN (Spanish Society for Natural Language Processing), TASS workshop has organized several tasks on Sentiment Analysis on Twitter in last years [5,11].

However, to determine the stance of group of people about a certain target, it is not enough to know if the opinions are positive or negative. To determine which opinion is the majority, it is necessary to know if the users are in favor or against the target. Note that a positive opinion is different than an opinion in favor, given that you can express positive opinions but against a target; for example, by praising the opposite position. Stance detection consist of automatically determining whether the author is in favor of the given target, against the target, or whether neither inference is likely. Sentiment Analysis can be done in a general way, but to carry out Stance detection properly, it is needed to know the target on which users express their opinions.

Moreover, knowing what type of users are participating in the discussion has great interest to conduct an adequate study of the opinions. In this regard, the goal of Author Profiling is to determine, from text, some characteristics of the author of that text. The most common characteristics that must be determined are gender and age.

Different international competitions have recently shown interest in Stance detection and Author Profiling: the task 6 at SemEval-2016 (Stance on Twitter) [14] that uses tweets in English and the Author Profiling task at PAN@CLEF 2016 [20] that uses texts written in English, Spanish, and Dutch extracted from several social networks.

Within the framework of the 33th conference of the SEPLN, it was organized the IberEval workshop that was dedicated to promoting the development of Human Language Technologies for Iberian languages. One of the share tasks proposed at IberEval was the Stance and Gender detection in Tweets on Catalan Independence (StanceCat) [23]. The aim of this task is to detect the author's gender and his/her stance with respect to the "independence of Catalonia" in tweets written in Spanish and Catalan.

In this paper, we present a system to address both Stance detection and User Profiling subtasks proposed at StanceCat task using the tweets written in Spanish. The system is based on the sequential representation of the tweets as input to a two-layer Convolutional Neural Network (CNN) [4] assembled with a final Long Short-Term Memory (LSTM) neural network [8]. Thus, it computes long-term relationships in the recurrent sub-network from short-term relationships computed in the convolutional sub-network.

A development phase was carried out to select the representation of the tweets that maximized the evaluation measure. The final representation was based on one-hot vectors at character level. Using the system described in this work, we achieved state-of-the-art results for the Stance detection subtask and the best results published until now for the Gender detection subtask for the Spanish language.

The rest of this paper is organized as follows. Section 2 presents StanceCat task and the corpus used in this paper. In Sect. 3, we present a short description of the system developed. Section 4 presents the experimental work conducted in this paper. Finally, in Sect. 5, we present some conclusions and the future work.

2 StanceCat at IberEval 2017

One of the tasks proposed in the 2017 edition of the IberEval workshop and in which more researchers participated was the StanceCat (Stance and Gender detection in Tweets on Catalan Independence) task. This task had a double objective. On the one hand, to determine the gender of the author of the tweet and on the other hand to determine the stance of the author, expressed in the tweet, with respect to the independence of Catalonia.

The StanceCat organizers acquired a corpus of 10800 tweets -5400 written in Spanish and 5400 written in Catalan- published between September and December 2015. All the tweets include the hashtag *#Independencia* or *#27S* to ensure that the target was the Catalan Independence. The corpus is labeled in terms of the gender of the author of each tweet (MALE or FEMALE) and in terms of the stance of the author respect to the independence of Catalonia (FAVOR, AGAINST, or NONE).

Although corpus includes tweets in Spanish and Catalan, we have only used those written in Spanish language. Table 1 shows the number of samples per each label in the Spanish subset of the StanceCat corpus. Note that the corpus is unbalanced in terms of Stance detection, being a clear bias between AGAINST and NONE with respect to FAVOR, that only represents the 7.8% of the samples. However, this unbalance does not occur in the Gender detection subtask where both labels are equally represented both in the training and the test sets.

Table 1. Number of samples per label in the Spanish subset of the StanceCat corpus.

	Training set			Test set		
	MALE	FEMALE	Total	MALE	FEMALE	Total
FAVOR	190	145	335	48	36	84
AGAINST	753	693	1446	188	173	361
NONE	1216	1322	2538	305	331	636
Total	2159	2160	4319	541	540	1081

The official evaluation measure for the Stance detection subtask was the macro-average of F_1(FAVOR) and F_1(AGAINST), without taking into account the NONE label. For the Spanish subtask, a total of 31 runs were submitted by ten participating teams. The results ranged from 48.88 to 19.06. The best result was obtained by the *iTACOS* [10] team with 48.88 of macro-averaged F_1. Authors used Support Vector Machine (SVM) as classification paradigm. One of the aspect to be highlighted of the *iTACOS* proposal is the features selection process. They define three types of features: stylistic features, structural features, and context features including the text of the webs linked in the tweet.

Regarding the Gender detection subtask, the official evaluation measure was the Accuracy. For the Spanish subtask, a total of 19 runs were submitted by five participating teams. The results ranged from 68.55 to 47.64. The organizers proposed a baseline based on the Low Dimensionality Representation approach. The best result was achieved by the *ELiRF-UPV* [6] team with 68.55 of Accuracy which was the only one team that exceeded the baseline. The *ELiRF-UPV* team tested several classification paradigm but the best result was obtained using SVM with bag-of-1grams and bag-of-2grams at character level.

More details about the task, the corpus, and the participants can be found in the review carried out by the task organizers and published in the IberEval workshop proceedings [23].

3 System Description

In this section we describe the main characteristics of the developed system to the Stance and Gender detection in Spanish tweets on Catalan Independence (StanceCat) task. This description includes the tweets preprocessing, their representation, and the system architecture.

3.1 Preprocessing and Representation of the Tweets

As a previous step, we performed a preprocessing of the tweets. We removed the accents and converted all the text to lowercase. Web links and numbers were substituted by two generic tokens (URL and NUMBER, respectively). We maintained *hashtags*, *emoticons* and *user mentions* and they where not substituted by a generic label as we did in previous works.

We tested several sequential representations of the tweets to model the order among the different units considered. Specifically, we used the following approaches:

– Embeddings (at word level): we considered a tweet x as a sequence of words, $x = x_1, x_2, ..., x_n$ and we represented each word x_i by means of its embedding vector $e(x_i) \in \mathbb{R}^{dw}$, where dw is the dimension of the word embedding.
– One-hot vectors (at char level): we considered a tweet x as a sequence of characters, $x = x_1, x_2, ..., x_n$ and we represented each character x_i by means of a one hot vector, $v(x_i) \in \mathbb{R}^{dc}$, where dc is the number of different characters in the corpus. To generate the one hot vectors, we only considered the characters that appeared in the training set.

Using these two representations, we can contrast the semantic information provided by the embeddings against the stylistic characteristics modeled by means of the one-hot representation and select the more relevant representation to the Stance and Gender detection tasks.

Regarding embeddings, in previous works, we used Word2Vec models [12,13] trained with the Spanish Wikipedia. In this work, we pre-trained our embeddings with 87 million tweets in Spanish. Our model is a skip-gram architecture. Each row of the lookup table has 300 dimensions and we used negative sampling as loss function.

Additionally, for the Stance detection task, we also used some emotion and polarity Spanish lexicons combined with the embeddings. In this case, each word x_i of a tweet was represented as a concatenation of its embedding $\mathrm{e}(x_i)$ with the information of the lexicons $\mathrm{l}(x_i)$. Specifically, these lexicons were:

- ElHPolar: a list of 1889 positive and 3301 negative words created from different resources [21].
- ISOL: a list of 2509 positive and 5626 negative words developed from Bing Liu's Opinion Lexicon [2]. This lexicon was automatically translated and manually reviewed [17].
- NRC Word-Emotion Association Lexicon: lexicon of 14182 words, automatically translated into Spanish. It contains information on polarity (negative, positive) and emotions (anger, fear, joy, sadness, disgust, trust, anticipation, and surprise) [15,16].
- MLSenticon: a lexicon composed of 11542 words that provides a polarity estimate in the interval [−1, 1] [1].

3.2 System Architecture

We explored different models depending on the representation of the tweets. This way, CNN [4] assembled with LSTM [8] were used to deal with sequential representations of tweets. These models computes a representation based on long-term relationships in the recurrent sub-network from short-term relationships computed in the convolutional sub-network. Finally, using this representation, a fully connected single-layer network with softmax activation functions computes the outputs of the network.

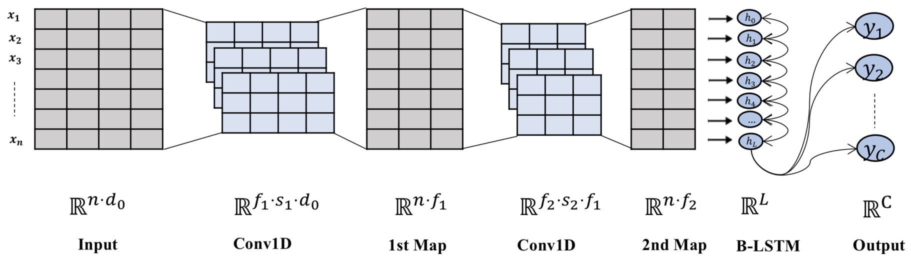

x_1
x_2
x_3
x_n
h_0
h_1
h_2
h_3
h_4
...
h_L
y_1
y_2
y_C
$\mathbb{R}^{n \cdot d_0}$
Input
$\mathbb{R}^{f_1 \cdot s_1 \cdot d_0}$
Conv1D
$\mathbb{R}^{n \cdot f_1}$
1st Map
$\mathbb{R}^{f_2 \cdot s_2 \cdot f_1}$
Conv1D
$\mathbb{R}^{n \cdot f_2}$
2nd Map
$\mathbb{R}^{L}$
B-LSTM
$\mathbb{R}^{C}$
Output

Fig. 1. System architecture.

Figure 1 shows a general scheme of the whole model, where n is the maximum number of elements in a tweet, d_0 is the dimensionality of the representation of each element, f_i is the number of filters in the convolutional layer i, s_i is the height of each filter in the layer i, L is the dimensionality of the *output state* of the LSTM, and C is the number of classes of the task.

This model receives as input a tweet represented as a matrix $M \in \mathbb{R}^{n \cdot d_0}$, where d_0 is denoted as dw at the word level, or dc at the character level. This input is directly passed to a CNN composed of two convolutional layers. For the Stance subtask, we set $f_1 = 8$, $f_2 = 16$ and $s_1 = s_2 = 3$. For the Gender detection subtask, this parameters were, $f_1 = 128$, $f_2 = 256$, $s_1 = s_2 = 3$. In both tasks padding was added in order to maintain the original number of rows. A MaxPooling layer, with $size = 2$, was applied to the output of the last convolutional layer. Later, a LSTM [8] was applied. For the Stance detection subtask we used an LSTM with $L = 64$ and for Gender detection subtask we used a Bidirectional LSTM [22] with $L = 2 * 128$. Finally, in both cases, we used the last output of the LSTM as input to a fully connected layer of C neurons, whose function is to perform the classification task.

To speed up the convergence and favor a correct training of the models, we used BatchNormalization [9] after each layer (including the input to directly normalize the training data) for the Gender detection subtask. We used noisy representations of training data, obtained after applying a Gaussian noise layer with $\sigma = 0.15$, with the aim of improving generalization [7]. We used $tanh$ activation functions after each normalized output with BatchNormalization except in the last layer where a $softmax$ activation function is used.

Finally, bucketing was used to train the models with variable length sequences. This allows us to build batches with representations of tweets that shares the same number of rows. In addition, the buckets were unsorted. This strategy was used both for reducing the training time and the over-fitting problem (similar to [3]).

4 Experimental Work

In this section we present the experimental evaluation of the systems introduced in Sect. 3. We report the results achieved both on the tuning and test phases.

4.1 Tuning Phase

In order to select the best representation and the best model for each task, a tuning process was performed. The training corpus provided by the organizers of the task was split into two sets, a set with the 80% of the tweets for training the model and the remaining 20% of the corpus was used as development set. These partitions were the same for all the tuning process.

Faced with the impossibility of testing all combinations of models and representations, only those combinations we thought that made more sense were considered.

Table 2. Tuning phase results for the Stance detection subtask.

Representation	$(F_{favor} + F_{against})/2$
Embeddings (Wikipedia)	51.84
Embeddings (Twitter)	52.19
Embeddings (Twitter) + Lexicons	51.08
One-hot (char level)	55.10

Table 2 shows the results on the development set for the Stance detection task. It can be observed how the embeddings, both of Wikipedia and Twitter, behave worse than the One-hot representation. In turn, the addition of emotion/polarity information through lexicons worsens the results.

Table 3. Tuning phase results for the Gender detection subtask.

Representation	Accuracy (%)
Embeddings Twitter	70.13 ± 3.05
One-hot (char level)	80.90 ± 2.62

Table 3 shows the results on the development set for the Gender detection task. It can be seen that, also in this subtask, the *One-hot* representation is the one that obtained the best results.

4.2 Test Phase

Once the best representation of the tweets for each subtask was chosen, we tested the final models on the official test set.

Table 4 shows the results achieved by the proposed system compared with the best system at StanceCat competition.

Table 4. Results for the Stance detection subtask.

System	$(F_{favor} + F_{against})/2$
Proposed system	46.37
Best system at StanceCat [10]	48.88

Although we can not overcome the best results of the competition team, we achieved the third place in the competition. We carried out an study of the performance of our system at class level. Table 5 shows this analysis in terms of *Precision*, *Recall*, and F_1 measure.

Table 5. Results at class level for Stance detection subtask.

Class	Precision	Recall	F_1
FAVOR	26.32	29.76	27.93
AGAINST	68.85	61.22	64.81
NONE	75.49	78.93	77.17

It can be seen that the worst results are obtained by the FAVOR class which is the class with minor number of samples, about 8% both in training and test sets. Our model did not include any mechanism to handle the imbalanced class problem. Note that the official evaluation measure do not take into account the NONE class in which we obtained the best results.

The results for Gender detection subtask, including confidence intervals, are shown in Table 6.

Table 6. Results for the Gender detection subtask.

System	Accuracy (%)
Proposed system	81.03 ± 2.33
Best system at StanceCat [6]	68.55 ± 2.76

We improved by 12.48 points the best previous result reported to this task [6] with the proposed architecture using one-hot vectors at character level as tweet representation. This improvement is statistically significant at 95% of confidence.

5 Conclusions and Future Work

In this paper, we presented a deep learning system for the Gender and Stance detection in Twitter. We explored different representation of the tweets - embeddings and one-hot vectors at character level- and an architecture that assembles CNN and LSTM neural networks.

We tested our approach in the Stance and Gender Detection in Tweets on Catalan Independence track at Ibereval 2017 workshop. With the proposed approach, we achieved state-of-the-art results for the Stance detection subtask and the best results published until now for the Gender detection subtask.

As future work, we plan to carry out a study of the obtained results. With this study, we will try to answer questions such as, why representations at the character level are more relevant than those based on embeddings for the addressed tasks? We plan to work in the development of techniques to handle the imbalanced classes, which has proved to be a very important problem in the Stance detection task.

We also plan to work with the subset of the corpus written in Catalan to verify the usefulness of the proposed system.

Acknowledgements. This work has been partially supported by the Spanish Ministerio de Ciencia, Innovación y Universidades and FEDER founds under AMIC project (TIN2017-85854-C4-2-R), and the Generalitat Valenciana under GUAITA project (INNVA1/2020/61).

The authors thank the organizers of the *Stance and Gender Detection in Tweets on Catalan Independence* track for provide us the StanceCat corpus.

References

1. Cruz, F.L., Troyano, J.A., Pontes, B., Ortega, F.J.: Building layered, multilingual sentiment lexicons at synset and lemma levels. Expert Syst. App. **41**(13), 5984–5994 (2014). http://www.sciencedirect.com/science/article/pii/S0957417414001997
2. Ding, X., Liu, B., Yu, P.S.: A holistic lexicon-based approach to opinion mining. In: Proceedings of the 2008 International Conference on Web Search and Data Mining, pp. 231–240. WSDM 2008, ACM, New York, NY, USA (2008). http://doi.acm.org/10.1145/1341531.1341561
3. Doetsch, P., Golik, P., Ney, H.: A comprehensive study of batch construction strategies for recurrent neural networks in MXNet. CoRR abs/1705.02414 (2017). http://arxiv.org/abs/1705.02414
4. Fukushima, K.: Neocognitron: a self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biol. Cybern. **36**(4), 193–202 (1980). https://doi.org/10.1007/BF00344251
5. García Cumbreras, M.A., et al.: Overview of TASS 2016. In: Proceedings of TASS 2016, pp. 13–21. CEUR Workshop Proceedings. CEUR-WS.org (2016)
6. González, J.A., Pla, F., Hurtado, L.F.: ELiRF-UPV at IberEval 2017: stance and gender detection in Tweets. In: Proceedings of the Second Workshop on Evaluation of Human Language Technologies for Iberian Languages (IberEval 2017), pp. 193–198. IberEval 2017, CEUR Workshop Proceedings. CEUR-WS.org (2017)
7. Grandvalet, Y., Canu, S., Boucheron, S.: Noise injection: theoretical prospects. Neural Comput. **9**(5), 1093–1108 (1997). http://www.mitpressjournals.org/doi/abs/10.1162/neco.1997.9.5.1093
8. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
9. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. CoRR abs/1502.03167 (2015). http://arxiv.org/abs/1502.03167
10. Lai, M., Cignarella, A.T., Hernández Farías, D.I.: iTACOS at IberEval2017: detecting stance in Catalan and Spanish Tweets. In: Proceedings of the Second Workshop on Evaluation of Human Language Technologies for Iberian Languages (IberEval 2017), pp. 185–192. IberEval 2017, CEUR Workshop Proceedings. CEUR-WS.org (2017)
11. Martínez Cámara, E., Díaz Galiano, M.C., García Cumbreras, M.A., Vega, M.G.: Overview of TASS 2017. In: Proceedings of TASS 2017, pp. 13–21. CEUR Workshop Proceedings. CEUR-WS.org (2017)
12. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. CoRR abs/1301.3781 (2013). http://arxiv.org/abs/1301.3781
13. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J.: Distributed representations of words and phrases and their compositionality. CoRR abs/1310.4546 (2013). http://arxiv.org/abs/1310.4546
14. Mohammad, S.M., Kiritchenko, S., Sobhani, P., Zhu, X., Cherry, C.: Semeval-2016 task 6: detecting stance in tweets. In: Proceedings of the International Workshop on Semantic Evaluation. SemEval 2016, San Diego, California, June 2016

15. Mohammad, S.M., Turney, P.D.: Emotions evoked by common words and phrases: using mechanical Turk to create an emotion lexicon. In: Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text, pp. 26–34. CAAGET 2010, Association for Computational Linguistics, Stroudsburg, PA, USA (2010). http://dl.acm.org/citation.cfm?id=1860631.1860635
16. Mohammad, S.M., Turney, P.D.: Crowdsourcing a word-emotion association lexicon. Comput. Intell. **29**(3), 436–465 (2013)
17. Molina-González, M.D., Martínez-Cámara, E., Martín-Valdivia, M.T., Perea-Ortega, J.M.: Semantic orientation for polarity classification in Spanish reviews. Expert Syst. App. **40**(18), 7250–7257 (2013). http://www.sciencedirect.com/science/article/pii/S0957417413004752
18. Nakov, P., Ritter, A., Rosenthal, S., Stoyanov, V., Sebastiani, F.: SemEval-2016 task 4: sentiment analysis in Twitter. In: Proceedings of the 10th International Workshop on Semantic Evaluation. SemEval 2016, Association for Computational Linguistics, San Diego, California, June 2016
19. Rosenthal, S., Farra, N., Nakov, P.: SemEval-2017 task 4: sentiment analysis in Twitter. In: Proceedings of the 11th International Workshop on Semantic Evaluation. SemEval 2017, Association for Computational Linguistics, Vancouver, Canada, August 2017
20. Rosso, P., Rangel, F., Potthast, M., Stamatatos, E., Tschuggnall, M., Stein, B.: Overview of PAN'16. In: Fuhr, N., et al. (eds.) CLEF 2016. LNCS, vol. 9822, pp. 332–350. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-44564-9_28
21. Saralegi, X., San Vicente, I.: Elhuyar at TASS 2013. In: XXIX Congreso de la Sociedad Espaola de Procesamiento de lenguaje Natural, Workshop on Sentiment Analysis at SEPLN (TASS2013), pp. 143–150 (2013)
22. Schuster, M., Paliwal, K.: Bidirectional recurrent neural networks. Trans. Sig. Proc. **45**(11), 2673–2681 (1997). https://doi.org/10.1109/78.650093
23. Taulé, M., Martí, M., Rangel, F., Rosso, P., Bosco, C., Patti, V.: Overview of the task of Stance and Gender Detection in Tweets on Catalan Independence at IBEREVAL 2017. In: Martínez, R., Gonzalo, J., Rosso, P., Montalvo, S., de Albornoz, J.C. (eds.) Notebook Papers of 2nd SEPLN Workshop on Evaluation of Human Language Technologies for Iberian Languages (IBEREVAL), pp. 157–177. CEUR Workshop Proceedings. CEUR-WS.org, 2017, Murcia, Spain, September 2017

Information Retrieval, Information Extraction

Natural-Annotation-Based Malay Multiword Expressions Extraction and Clustering

Wuying Liu[1] and Lin Wang[2](✉)

[1] Laboratory of Language Engineering and Computing, Guangdong University of Foreign Studies, Guangzhou 510420, Guangdong, China
wyliu@gdufs.edu.cn

[2] Xianda College of Economics and Humanities, Shanghai International Studies University, Shanghai 200083, China
lwang@xdsisu.edu.cn

Abstract. Multiword expression (MWE) is an optimal granularity of language reuse. However, no explicit boundaries between MWEs and other words causes a serious problem on automatic identification of MWEs for some less commonly taught languages. This paper addresses the issue of Malay MWEs extraction and clustering, and proposes a novel unsupervised extraction and clustering algorithm based on natural annotations. In our algorithm, we firstly use a binary classification for each space character to solve length-varying Malay MWEs extraction, secondly transfer natural document-level category annotations to MWE-level ones for Malay MWEs clustering, and finally distill out a general MWEs resource and several domain resources. The experimental results in the Malay dataset of 272,783 text documents show that our algorithm can extract MWEs precisely and dispatch them into domain clusters efficiently.

Keywords: Multiword expression · Natural annotation · Extraction · Clustering · Malay

1 Introduction

Language granularity has always been a difficult puzzle for Machine Translation [1] and Natural Language Processing. A multiword expression (MWE) is a lexeme made up of a sequence of two or more lexemes and its meaning cannot be obtained from its parts. This MWE granularity, between words and sentences, will be a more suitable size for a language reuse efficiently [2].

The MWEs extraction for common languages [3, 4] has been widely investigated since the early days of Natural Language Processing, and many rule-based or statistic-based algorithms have been proposed [5, 6]. However, the investigation on the MWEs extraction for less commonly taught languages is still rare now.

Modern Malay (Bahasa Melayu), a less commonly taught language, is spoken by 290 million people in Brunei, Indonesia, Malaysia, and Singapore, whose alphabet is just the same to that of English. The fast-paced development of linguistic science and

A. Gelbukh (Ed.): CICLing 2018, LNCS 13396, pp. 143–152, 2023.
https://doi.org/10.1007/978-3-031-23793-5_13

computing technology has led to the accumulation of large-scale Malay text documents on the Internet in recent years. For instance, the Wikipedia contains more than 300 thousand Malay text documents today.

This explosion of language big data with natural annotations has brought a great opportunity to Less Commonly Taught Language Processing, not only MWEs extraction but also MWEs clustering. This paper addresses the issue of Malay MWEs extraction and clustering, and tries to build a general Malay MWEs resource and several domain resources automatically.

2 Related Work

There have been many MWEs extraction algorithms proposed for common languages up to the present. The early supervised algorithm uses statistical context features to reach a relatively high accuracy on MWEs extraction for common languages [7]. Subsequently, the semi-supervised algorithm uses morphosyntactic features to create Arabic verbal MWEs resource [8]. The previous supervised and semi-supervised algorithms normally require manual annotations, and this procedure will be time-consuming and expensive, which may not keep up with the rapid expansions of big data.

Recent investigations have used an unsupervised algorithm to extract Vietnamese multisyllabic words [9], whose scientific problem is very similar to that faced by MWEs extraction. The unsupervised algorithm is a straightforward and efficient model for real incremental and online applications of big data, whose effectiveness is due to that simple models and a lot of data trump more elaborate models based on less data in contemporary big data era [10]. Unsupervised learning does not require any manual annotation [11], which will be more suitable for MWEs extraction from dynamic language big data.

Some researchers used mutual information and information entropy to extract Chinese MWEs from Internet news pages and obtained an excellent result [12]. Some investigations defined the tokenization issue of the less commonly taught language as a binary classification algorithm for each space character to identify the token boundaries [13], which motivated us to implement MWEs extraction according to a binary classification algorithm. There have been also some studies to implement fine-grained instance-level annotating using multi-instance learning from coarse-grained document-level annotations [14], which has important reference significance for our MWEs clustering.

Based on the above motivations from related works, we design a novel unsupervised architecture, which takes full advantage of language big data and straightforward efficient model and uses a binary classification algorithm for each space character to implement MWEs extraction, and which makes full use of document-level natural annotations to implement MWEs clustering.

3 Architecture

Figure 1 shows our unsupervised extracting and clustering architecture for Malay MWEs extraction and clustering, which mainly includes two parts: **Extracting** and **Clustering**. The **Extracting** part receives large-scale Malay text documents and produces Malay

MWEs, and the **Clustering** part receives the Malay MWEs generated by the **Extracting** part and outputs a general MWEs and several domain MWEs, which can be used as general and domain lexical resources for Natural Language Processing. From this global perspective, the architecture is a meta-level structure and various suitable extracting and clustering algorithms can be implemented within it.

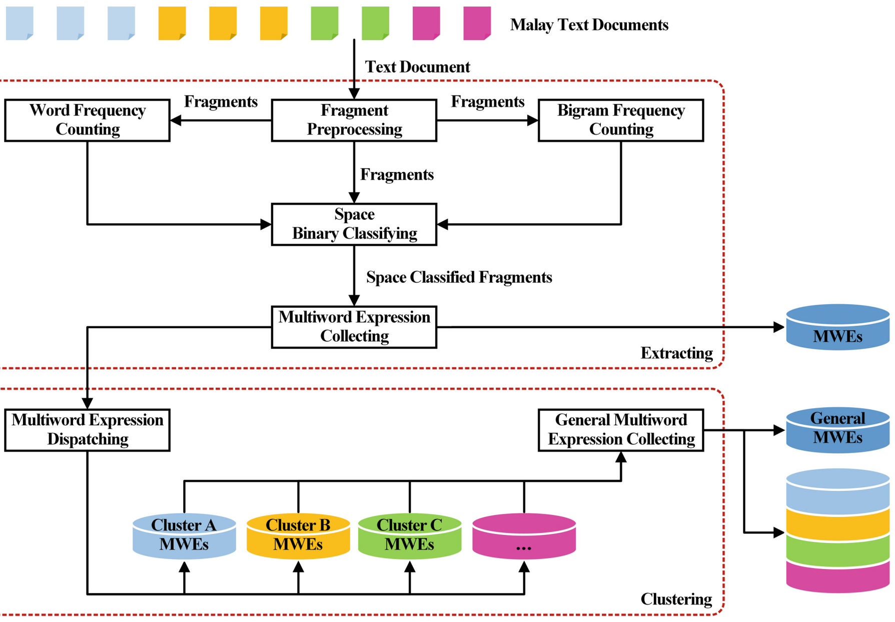


Fig. 1. Unsupervised extracting and clustering architecture.

In raw Malay texts, space character, similar to that in Vietnamese, can also be regarded as an overload character: a connector within a MWE or a separator between MWEs. Based on this understanding, we define the Malay MWEs extraction as a binary classification task for each space character in the **Extracting** part. There are total five processing units to complete the extracting work together. When a Malay text document arrives, the ***Fragment Preprocessing*** unit will be triggered firstly to split the text into several fragments, which are pure Malay word sequences without any punctuations, English words and other characters. Subsequently, the ***Word Frequency Counting*** unit and the word-level ***Bigram Frequency Counting*** unit receive the generated fragments respectively and count the corresponding frequency in parallel. The ***Space Binary Classifying*** unit receives the fragments and the unsupervised knowledges of word frequency and bigram frequency, and uses our proposed connectivity-based method to classify each space character: if a space character is a connector within a MWE, the method will output a dash character ('-') to replace it; if it is a separator between MWEs, the method will maintain it as a space character (' ') in the classified fragments. Finally, the ***Multiword Expression Collecting*** unit tokenizes all space classified fragments only by space characters and counts the frequency of each candidate MWE token, at least containing one dash character. According to the statistical results of frequency and a preset frequency threshold (it is 3 in this paper), the unit will collect a set of MWEs.

The effectiveness of above binary classifying method depends on the amount of Malay text documents. Fortunately, the explosion of language big data from Internet brings a lot of Malay news pages, which can be crawled automatically to form a large-scale dataset of raw text documents. Furthermore, most of news pages normally have manual category annotations. These natural document-level category annotations can be transformed down to MWE-level cluster annotations in the **Clustering** part, where there are only two processing units. The ***Multiword Expression Dispatching*** unit incrementally creates MWE clusters named from category annotations of news text documents, and dispatches each extracted MWE into a related cluster according to the category annotation of the text document containing the MWE. The ***General Multiword Expression Collecting*** unit selects those MWEs with a high cluster frequency to form a set of general MWEs. At the same time, the unit removes the general MWEs from each cluster, and distills out multiple domain MWE sets.

4 Algorithm

Within the unsupervised extracting and clustering architecture, we design a detailed Malay MWEs extracting and clustering algorithm, which is shown in Fig. 2.

The ***extracting*** function is the pseudo-code implementation of the **Extracting** part of our above architecture. While the ***clustering_general*** function and the ***clustering_domain*** function together implement the **Clustering** part. The main contribution of the algorithm is the computing definition of the connectivity (line 22 in Fig. 2). We regard the co-occurrence probability to individual occurrence probability of two neighboring words as a connectivity degree of the space character between the two words. This idea can avoid the non-fixed number of words problem in MWEs extraction. So, our algorithm can complete length-varying Malay MWEs extraction only by counting word frequency and bigram frequency. In our algorithm, we also make full use of document-level natural annotations to implement MWEs clustering straightforwardly. In the ***clustering_general*** function, we collect those MWEs occuring in more than 60% clusters to form the general MWEs.

During the total running process of the algorithm, the space overhead is mainly used to cache two indexes of word-frequency map and bigram-frequency map, which is negligible for the current 64bits memory capacity. Though the main time overhead is proportional to the size of the Malay text documents. The stream processing method, only one-pass scanning of each text document for two indexes construction, will make it time-efficient. The space-time complexity of the Malay MWEs extracting and clustering algorithm is acceptable in practical applications.

```
// Malay MWEs Extracting and Clustering Algorithm
Input:  Document[] mtds; // Malay Text Documents
        Float ct; // Connectivity Threshold
Output: String[] mwes; // Multiword Expressions
        String[] gmwes; // General Multiword Expressions
        Cluster[] dmwes; // Domain Multiword Expressions
Function String[]: extracting(Document[] mtds; Float ct)
String[] frags;
Map<String, Integer> bf;
Map<String, Integer> wf;
For Integer i ← 1 To mtds.size Do
        String[] frag ← fragmentpreprocessing(mtds[i].text);
        frags.merge(frag);
        bf.merge(bigramfrequencycounting(frag));
        wf.merge(wordfrequencycounting(frag));
End For
// Space Binary Classifying
For Integer j ← 1 To bf.keyset.size Do
        Integer bwf ← bf.get(bf.keyset[j]);
        Integer fwf ← wf.get(bf.keyset[j].firstword);
        Integer swf ← wf.get(bf.keyset[j].secondword);
        Float c ← (bwf/fwf+bwf/swf)/2;
        If (c>ct)
        Then frags.update(bf.keyset[j].firstword+'-'+bf.keyset[j].secondword);
        End If
End For
mwes ← multiwordexpressioncollecting(frags);
Return mwes.

Function String[]: clustering.general(Document[] mtds; String[] mwes)
// Multiword Expression Dispatching
For Integer i ← 1 To mwes.size Do
        Document[] docs ← mtds.getdocuments(mwes[i]);
        For Integer j ← 1 To docs.size Do
                String category ← docs[j].category;
                If (dmwes.contain(category))
                Then dmwes.get(category).add(mwes[i]);
                Else dmwes.put(Cluster.new(category).add(mwes[i]));
                End If
        End For
End For
gmwes ← collectinggeneral(dmwes);
Return gmwes.

Function Cluster[]: clustering.domain(String[] gmwes; Cluster[] dmwes)
For Integer i ← 1 To dmwes.size Do
        dmwes[i].remove(gmwes);
End For
Return dmwes.
```

Fig. 2. Malay MWEs extracting and clustering algorithm.

5 Experiment

In order to validate the effectiveness of our unsupervised extracting and clustering architecture and algorithm, we firstly prepare large-scale Malay text documents by crawling news pages on the Internet and gather total 272,783 Malay plain-text documents with natural category annotations. Furthermore, we already have a list with 85,539 Malay MWEs, which will be used as the golden standard to evaluate the experimental results. Secondly, we implement our unsupervised extracting and clustering algorithm, and run extracting and clustering functions respectively in the above dataset. We also implement another unsupervised extracting algorithm according to mutual information and information entropy as the baseline [12]. Finally, we analyze the experimental results and suggest some discussions.

5.1 Result and Discussion About Extraction

In the unsupervised extracting part of experiments, we run our algorithm under different connectivity threshold from 0.1 to 0.9 and report the classical Precision (P), Recall (R) and F1-measure (F1) to evaluate the result of extraction. Figure 3 shows the detailed trends of the three measures. We can find that the P values and the R values have contrary trends with the increase of the connectivity threshold. When the connectivity threshold equals 0.3, the F1 value will climb the optimal peak, where the P, R and F1 values are 0.2965, 0.2311 and 0.2597 respectively. We also run the baseline in the same environment and gain its best P, R and F1 values (0.0693, 0.3013 and 0.1126). The experimental results

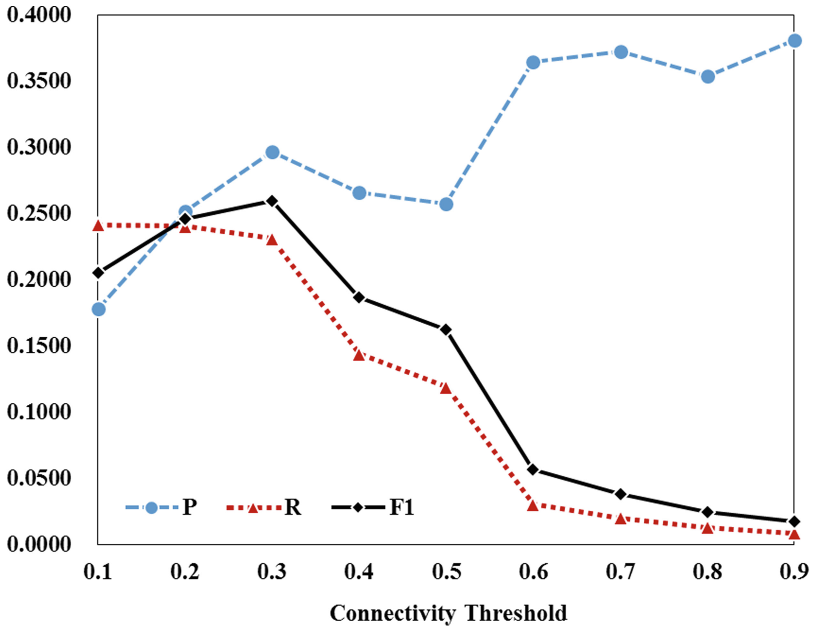


Fig. 3. Evaluation result of extraction.

prove that the performance of our straightforward unsupervised extracting is superior to that of mutual information and information entropy method.

Though the experimental extracting precision is only about 30% under the situation of the best F1 value. The main reason is that the Malay text documents are independent on the golden standard. The actual extracting precision will high exceed the experimental result. When the F1 value equals the best 0.2597, we can obtain 66,625 Malay MWEs. Table 1 shows the partial examples of the extracted Malay MWEs, from which linguists estimate that the actual extracting precision will over 80%.

Table 1. Partial examples of the extracted malay MWEs.

adat resam	anak watan	ayam pedaging	ARKADIUSZ Milik
adu domba	anak yatim	ayam piru	ASCOLI PICENO
aedes albopictus	anak yatim piatu	ayam tambatan	ATHLETIC BILBAO
ah long	angin ahmar	AARON Aziz	AUNG San Suu Kyi
ahlan wasahlan	angkasa lepas	ABDULLA Yameen	AYDA Jebat
ahli nujum	angkat sumpah	ADELINE Tsen	AYER KEROH
air kencing	angkatan tentera	AFRIKA SELATAN	AZ Alkmaar
air liur	anjakan paradigma	AHMAD NISFU	AZREL Ismail
air mani	anjing penghidu	AIDE Iskandar	Aamir Khan
air pancut	anugerah Grammy	ALEXANDRE Pato	Aaron Ago Dagang
air pasang	apam balik	ALI HAMSA	Aaron Cresswell
air zamzam	apit kanan	ALOR SETAR	Abby Fana
akad nikah	arang batu	AMELIA ALICIA ANSCELLY	Abdelaziz Bouteflika
akan datang	asam garam	AMERIKA SYARIKAT	Abdoulaye Faye
akar umbi	asid benzoik	AMPANG JAYA	Abdul Azeez Abdul Rahim
akil baligh	asid deoksiribonukleik	AMYRA Rosli	Abdul Gani Patail
alam barzakh	asid folik	ANDORRA LA VELLA	Abdul Ghani Minhat
alam sekitar	asid hidroklorik	ANGKAT BERAT	Abdul Halim
alam semesta	asid nitrik	ANGKATAN Bersenjata	Abdul Hamid

(*continued*)

Table 1. (*continued*)

alat penimbang	asid sulfurik	ANNUAR Musa	Abdul Hamid Pawanteh
amar makruf nahi mungkar	asid urik	ANTHONY Kevin Morais	Abdul Khaliq Hamirin
anak bongsu	awan kumulonimbus	ANTOINE Griezmann	Abdul Muntaqim
anak didik	awan kumulus	ANUGERAH PLANET MUZIK	Abdul Rahman Dahlan
anak panah	awet muda	ANWAR ibrahim	Abdul Rahman Palil
anak pinak	ayam golek	ARITZ Aduriz	Abdul Taib Mahmud

5.2 Result and Discussion About Clustering

In the unsupervised clustering part of experiments, we run our algorithm to cluster the above 66,625 Malay MWEs according to their original category annotations, and finally we select out a general resource with 3,419 Malay MWEs and distill out total 9 domain resources. The detailed number of Malay MWEs in each domain cluster is shown in Table 2.

Table 2. Number of malay MWEs.

General MWEs	Domain MWEs	
Number	Cluster annotation	Number
3,419	dunia	8,284
	jenayah	5,733
	nasional	16,023
	semeasa	10,797
	sukan	13,531
	utusan borneo-berita iban	14,906
	utusan borneo-berita nasional	12,739
	utusan borneo-berita sarawak	20,530
	utusan borneo-sukan	7,628

The experimental results show that the unsupervised clustering function can transmit natural document-level category annotations to MWE-level ones efficiently, and support online incremental construction. If you want to obtain more fine-grained domain cluster, you can cluster the Malay text documents first and then use our unsupervised clustering function.

6 Conclusion

This paper proposes a natural-annotation-based MWEs extraction and clustering algorithm for Malay general and domain resources construction. The experimental results show that the effectiveness of our algorithm only depends on the context knowledge of co-occurrence frequency of words and natural document-level category annotations.

Further research will concern the influence of additional language knowledge such as word formation, stop word, syntax, and even semantics. We will also transfer above research productions to other suitable Austronesian languages like Indonesian, Filipino, and so on.

Acknowledgements. The research is supported by the Social Science Foundation of Shanghai (No. 2019BYY028), the Humanity and Social Science Research Project of Ministry of Education of China (No. 20YJAZH069, No. 20YJC740062), and the Guangdong Basic and Applied Basic Research Foundation (No. 2018A030313672).

References

1. Aw, A., Aljunied, S.M., Li, H.: Malay multi-word expression translation. In: Proceedings of the Workshop on Technologies and Corpora for Asia-Pacific Speech Translation (2009)
2. de Caseli, H.M., Ramisch, C., das Graças Volpe Nunes, M., Villavicencio, A.: Alignment-based extraction of multiword expressions. Language Resources and Evaluation **44**(1), 59–77 (2010). https://doi.org/10.1007/s10579-009-9097-9
3. Hore, C., Asahara, M., Matsumoto, Y.: Automatic extraction of fixed multiword expressions. In: Dale, R., Wong, K.-F., Su, J., Kwong, O.Y. (eds.) IJCNLP 2005. LNCS (LNAI), vol. 3651, pp. 565–575. Springer, Heidelberg (2005). https://doi.org/10.1007/11562214_50
4. Weller, M., Heid, U.: Extraction of German multiword expressions from parsed corpora using context features. In: Proceedings of the 7th International Conference on Language Resources and Evaluation, pp. 3195–3201 (2010)
5. Attia, M., Toral, A., Tounsi, L., Pecina, P., Van Genabith, J.: Automatic extraction of Arabic multiword expressions. In: Proceedings of the Multiword Expressions: From Theory to Applications, pp. 19–27 (2010)
6. Dubremetz, M., Nivre, J.: Extraction of nominal multiword expressions in French. In: Proceedings of the 10th Workshop on Multiword Expressions, pp. 72–76 (2014)
7. Farahmand, M., Martins, R.: A supervised model for extraction of multiword expressions based on statistical context features. In: Proceedings of the 10th Workshop on Multiword Expressions, pp. 10–16 (2014)
8. Al-Badrashiny, M., Hawwari, A., Ghoneim, M., Diab, M.: SAMER: a semi-automatically created lexical resource for Arabic verbal multiword expressions tokens paradigm and their morphosyntactic features. In: Proceedings of the 12th Workshop on Asian Language Resources, pp. 113–122 (2016)
9. Liu, W., Wang, L.: Unsupervised ensemble learning for Vietnamese multisyllabic word extraction. In: Proceedings of the 20th International Conference on Asian Language Processing, pp. 353–357 (2016)
10. Halevy, A., Norvig, P., Pereira, F.: The unreasonable effectiveness of data. IEEE Intell. Syst. **24**(2), 8–12 (2009)
11. Vlachos, A.: Evaluating unsupervised learning for natural language processing tasks. In: Proceedings of the 1st Workshop on Unsupervised Learning in NLP, pp. 35–42 (2011)

12. Liu, J., Tang, H., Liu, W.: An extraction method for Chinese terminology based on statistical technology. China Terminology **16**(5), 10–14 (2014)
13. Liu, W.: Supervised ensemble learning for Vietnamese tokenization. Internat. J. Uncertain. Fuzziness Knowl.-Based Syst. **25**(2), 285–299 (2017)
14. Zhang, Y., Surendran, A.C., Platt, J.C., Narasimhan, M.: Learning from multi-topic web documents for contextual advertisement. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1051–1059 (2008)

Self-adaptive Privacy Concern Detection for User-Generated Content

Xuan-Son Vu(✉) and Lili Jiang

Department of Computing Science, Umeå University, Umeå, Sweden
{sonvx,lili.jiang}@cs.umu.se

Abstract. To protect user privacy in data analysis, a state-of-the-art strategy is differential privacy in which scientific noise is injected into the real analysis output. The noise masks individual's sensitive information contained in the dataset. However, determining the amount of noise is a key challenge, since too much noise will destroy data utility while too little noise will increase privacy risk. Though previous research works have designed some mechanisms to protect data privacy in different scenarios, most of the existing studies assume uniform privacy concerns for all individuals. Consequently, putting an equal amount of noise to all individuals leads to insufficient privacy protection for some users, while over-protecting others. To address this issue, we propose a self-adaptive approach for privacy concern detection based on user personality. Our experimental studies demonstrate the effectiveness to address a suitable personalized privacy protection for cold-start users (i.e., without their privacy-concern information in training data).

Keywords: Multi-task learning · Privacy detection · User-generated content · Differential privacy · Personality profiling

1 Introduction

Recent advances in artificial intelligence (AI) have opened many new possibilities (e.g., data analytics, autonomous systems), however, they pose the risk regarding privacy and security [1]. Together with the advances in AI and the increasing volume of user-generated content (UGC) such as social network data and medical data, privacy concern on personal data becomes even more critical [2]. Especially, the cross-disciplinary studies have been conducted with the need of integrating personal data from multiple sources (e.g., studies by combining diet data, work condition data, and health data). This data integration dramatically increases the risk of privacy leakage. For example, Narayanan et al. [3] de-anonymized the published Netflix Prize data by matching with IMDB data[1]. Moreover, not only connecting to external sources will reveal privacy but also internal model parameters. Fredrikson et al. [4] used hill-climbing algorithm on the output probabilities of a computer-vision classifier to reveal individual faces

[1] http://www.imdb.com/.

A. Gelbukh (Ed.): CICLing 2018, LNCS 13396, pp. 153–167, 2023.
https://doi.org/10.1007/978-3-031-23793-5_14

from the training data. As witnessed by these demonstrations and because privacy guarantees must apply to the worst-case outliers (not only the average), any strategy for protecting data privacy should prudently assume that attackers have unfettered access to external data sources as well as internal model parameters.

UGC data has been used in many research areas ranging from psychology (e.g., predicting personality [5,6]) to data analytics (e.g., predicting stock market [7]). The privacy issues become more and more critical, especially when sensitive information (e.g., age, gender) can be derived from UGC data [8]. Thus, the main goal of this paper is to present a self-adaptive approach for privacy concern detection, which automatically detects the privacy need of individuals based on personality information extracted from their UGC data. In this way, we provide trade-off of sufficient privacy protection and data utility. The **main contributions** of this paper include:

– Introducing a neural network model that can learn and automatically predict the privacy-concern degree of individuals based on their personalities.
– Evaluating the effectiveness of personality based privacy-guarantee through extensive experimental studies on a real UGC dataset.
– Solving an imbalanced data distribution issue in privacy-concern detection raised by Vu et al. [9] using an over-sampling approach.

The remainder of this paper is organized as follows: In Sect. 2, we present related work, and introduce Differential Privacy [10] and the Five Factor Model [11]. Our proposed methodology is presented in Sect. 3. The experimental methodology is explained in Sect. 4. Experimental result analysis and discussion are in Sect. 5. Section 6 concludes the paper and presents future work.

2 Related Work

Anonymization [12] and sanitization [13] have been widely used in privacy protection. Differential privacy [10] later emerged as the key privacy-guarantee mechanism by providing rigorous, statistical guarantees against any inference from an adversary. Based on differential privacy, some privacy-oriented frameworks arose including PINQ [14] and GUPT [15]. Moreover, they use a unified amount of noise for privacy protection. To overcome the disadvantage of injecting uniform noise in differential privacy, recent works [16,17] have proposed personalized differential privacy methods, in which they apply different amounts of noise on different users. However, the limitation is that they decided privacy budget by either random sampling or query involvements regardless of the provenance individual's actual privacy concern. Particularly, Jorgensen et al. [17] use random sampling for personalized DF and Hamid et al. [16] consider what records the given query involves to decreases the corresponding privacy budgets. Thus, we propose a personality-based differential privacy approach in a self-adaptive way to calculate privacy concern for reasonable privacy protection and data utility.

2.1 Differential Privacy Preliminaries

Differential privacy (DP) [10] has established itself as a strong standard for privacy preservation. It provides privacy guarantees for algorithms analyzing databases, which in our case is a machine learning algorithm processing a training dataset and histogram-based data analysis. The key idea behind differential privacy is to obfuscate an individual's properties, but not the whole group's properties in a given database. So the probability for any individual in the database needs to have a property that should barely differ from the base rate (i.e., the chance to guess whether an individual is involved in one study or not). When an attacker analyzes the database, he/she cannot reliably learn anything new about any individual in the database, no matter how much additional information he/she has. The following is a formal definition of (ϵ-δ) differential privacy. We assume a database D consisting of n vectors of m-components over some set $\mathcal{F}$ represented as a $m \times n$ matrix over $\mathcal{F}$.

Definition 1 (Distance Between Databases). Define

$$\operatorname{dist}(D, D') := |\{i \in \{1, 2, \ldots, m\}\colon D_i \neq D'_i\}| \forall D,\ D' \in (\mathcal{F}^m)^n$$

as the number of entries in which the databases D and D' *differ.* Differential privacy is defined using pairs of adjacent databases in present work, which only differ by one record (i.e., $\operatorname{dist}(D, D') = 1$).

Definition 2 (Probability Simplex). Given a discrete set B, the probability simplex over B, denoted $\Delta(B)$ is defined to be:

$$\Delta(B) = \left\{x \in \mathbb{R}^{|B|} : x_i \geq 0 \text{ for all } i \text{ and} \textstyle\sum_{i=1}^{|B|} x_i = 1\right\}$$

A randomized algorithm with domain A and (discrete) range B will be associated with a mapping from A to the probability simplex over B, denoted as $\Delta(B)$.

Definition 3 (Randomized Algorithm). A randomized algorithm $\mathcal{M}$ with domain A and discrete range B is associated with a mapping $\mathcal{M} : A \rightarrow \Delta(B)$. On input $a \in A$, the algorithm $\mathcal{M}$ outputs $\mathcal{M}(a) = b$ with probability $(\mathcal{M}(a))_b$ for each $b \in B$.

Definition 4 ((ϵ-δ)-differential privacy). Let $\mathcal{M}$ be a randomized algorithm processing D and Range($\mathcal{M}$) its image. Now $\mathcal{M}$ is called (ϵ-δ)-differentially private if $\forall \mathcal{S} \subseteq \text{Range}(\mathcal{M})$:

$$\forall D, D'\colon \operatorname{dist}(D, D') \leq 1 \Rightarrow \Pr\left[\mathcal{M}(D) \in \mathcal{S}\right] \leq e^{\epsilon} \cdot \Pr\left[\mathcal{M}(D') \in \mathcal{S}\right] + \delta$$

Intuitively, differential privacy controls the degree to which D and D' can be distinguished. When $\delta = 0$ then (ϵ-δ)-differential privacy is also called ϵ-differential privacy. Smaller ϵ gives more privacy and lower utility. Then, given the result of a randomized algorithm $\mathcal{M}$, an attacker cannot learn any new property about data subjects with a significant probability.

The Global Privacy Budget. PINQ [14] is an implementation of interactive differential privacy which ensures, at runtime, that queries adhere to a global privacy budget ϵ. Its central principle is that multiple queries (e.g., with differential privacy ϵ_1 and ϵ_2 respectively) have an additive effect $\epsilon_1 + \epsilon_2$ on the overall differential privacy. In other words, for a dataset queried q times, with each query having privacy parameter ϵ_i, the total privacy budget of the dataset is given by $\epsilon_{total} = \sum_{i=1}^{q} \epsilon_i$. PINQ also tracks sensitivity of functions to track how much to deduct from the global privacy budget on each invocation of a primitive query. As mentioned in [16], the global privacy budget has limitations when applied to an interactive system: (1) data analysts using the system, may run out of privacy budget even before obtaining valuable results and (2) a global budget is not capable of handling a live database when new records are frequently added.

2.2 The Five Factor Model

Regarding personality prediction, the most influential Five Factor Model (*FFM*) has become a standard model in psychology over the last 50 years [11]. Here we re-introduce a summary of Vu et al. [9] regarding FFM. The five factors are defined as *neuroticism, openness to experience, conscientiousness, agreeableness*, and *extraversion*. Pennebaker et al., [5] identify many linguistic features associated with each of personality traits in *FFM*. (1) *Extroversion* (sEXT) tends to seek stimulation in the external world, the company of others, and to express positive emotions. (2) *Neurotics* (sNEU) people use more 1st person singular pronouns, more negative emotion words than positive emotion words. (3) *Agreeable* (sAGR) people express more positive and fewer negative emotions. Moreover, they use relatively fewer articles. (4) *Conscientious* (sCON) people avoid negations, negative emotion words and words reflecting discrepancies (e.g., should and would). (5) *Openness to experience* (sOPN) people prefer longer words and tentative expressions (e.g., perhaps and maybe), and reduce the usage of 1st person singular pronouns and present tense forms.

3 Methodology

As mentioned above, one limitation of differential privacy is the unified privacy budget on all individuals in the same dataset. To address this limitation, we propose a personality-based differential privacy algorithm. The proposed approach is motivated by the findings on the statistically verified correlation between personality and privacy concerns of individuals on Facebook [18]. In their work, they output the correlation values as *p-value* $= \{.003, .007, .010\}$ for {cNEU, cEXT, cAGR} personality traits accordingly. Sumner et al., however, did not mention about the *p-value* of the correlation between privacy-concern and {cCON, cOPN} personal traits. Moreover, this personality-based privacy can be characterized as personalized-differential privacy that also satisfies ϵ-differential privacy by the proof of Ebadi et al. [16]. However, Ebadi et al. did not have an automatic way of detecting personalized privacy-concern level, which we are addressing.

Problem Definition

Given a database $\mathcal{D}$ consists of N user records $U = \{T, P\}$, where T is a set of textual features and P is a set of five personality trait scores in the Five Factor Model (FFM) [19]. Our target is predicting a real value r that represents the user privacy-concern degree. The r value, later on, will be used to decide the amount of noise and privacy-budget of the user to protect her/his data privacy more reasonably.

A Baseline Linear Regression Model

As a baseline linear regression model, we learn a linear function $y = w * \bar{x} + b$ where $\bar{x}$ is an input vector and b is a bias value. Given a real outcome $\hat{y}$, we can calculate a loss function $\mathcal{L} = \frac{1}{2}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$, where N is the number of samples and use it for the optimization process to find the best values of the weighting matrix w, i.e., $w^* = arg \min_{w} \mathcal{L}(w)$. Since we have five personality scores, therefore, five linear regression models are learned to predict scores of the five personality traits (i.e., *neuroticism, openness to experience, conscientiousness, agreeableness, and extraversion*). Lastly, we directly adopt a scaled weighting vector $V = (5.0, 4.7, 4.3, 4.1, 1.0)$ from [18] to calculate $r = sigmoid(z)$, where $z = \sum_{i=1}^{5} v_i * y_j$, $V = (v_i \mid i = 1, 2 \ldots, 5)$ and $Y = (y_j \mid j = 1, 2 \ldots, 5)$ is the output vector when predicting the five personality traits.

A Deep Neural Network Regression Model

We hypothesize that the five personality traits are highly correlated, therefore, a jointly learning regression model will lead to a better prediction model compared to the five standalone linear regression models. Thus, differently from the baseline, we design a method, a multilayer perceptron neural network model (MLP) [20] to learn and predict r value directly (see Fig. 1).

Formally, a one-hidden-layer MLP is a function $f : R^K \rightarrow R^L$, where K is the size of input vector x and L is the size of the output vector $f(x)$, such that, in matrix notation: $f(x) = G(b^{(2)} + W^{(2)}(s(b^{(1)} + W^{(1)}x)))$, with bias vectors $b^{(1)}, b^{(2)}$; weight matrices $W^{(1)}, W^{(2)}$ and activation functions G and s. The vector $h(x) = \Phi(x) = s(b^{(1)} + W^{(1)}x)$ constitutes the hidden layer. $W^{(1)} \in R^{K \times K_h}$ is the weight matrix connecting the input vector x to the hidden layer h. Each column $W_{\cdot i}^{(1)}$ represents the weights from the input units to the *i-th* hidden unit. The output vector is then obtained as: $o(x) = G(b^{(2)} + W^{(2)}h(x))$. To train

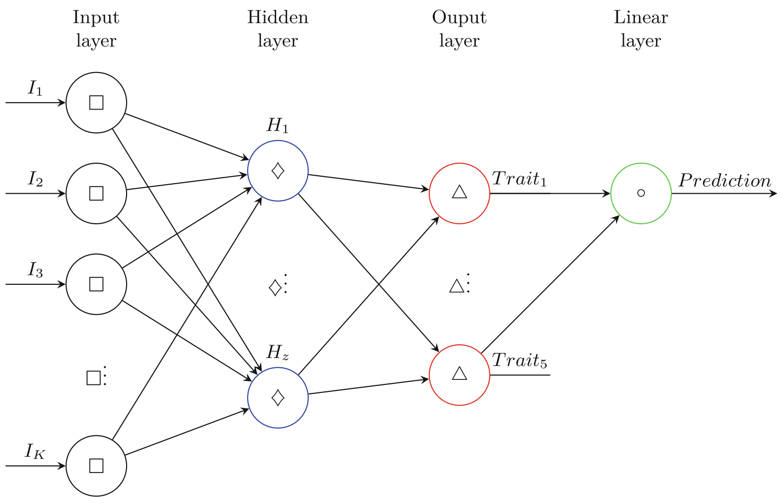


Fig. 1. Network graph of a multilayer perceptron with K input units, L output units (L=5), and a linear layer.

an MLP, we learn all parameters of the model, and here we use Adam optimizer [21] with minibatches to control the learning rate. The set of parameters to learn is the set $\theta = \{W^{(2)}, b^{(2)}, W^{(1)}, b^{(1)}\}$. The gradients $\partial\ell/\partial\theta$ can be obtained through the backpropagation algorithm [22] (a special case of the chain-rule of derivation). Lastly, in the linear layer, we apply a similar approach to the baseline linear regression method to calculate the final prediction.

Implementation Details: we use Tensorflow [23] to implement our model. Model parameters are learned to minimize the cross-entropy loss with L_2 regularization. Table 1 shows optimal settings for the regression task and Fig. 2 shows the learning curve of the MLP model on training data following 10-fold cross-validation schema (i.e., each fold, 10% is used for validation).

Table 1. The optimal hyperparameter settings.

Name	Value
Hidden layers	80
# epoch	90
Learning rate	10^{-5}
l_2	10^{-5}

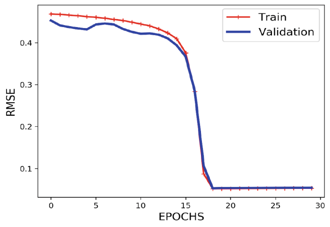


Fig. 2. Learning curve of the MLP.

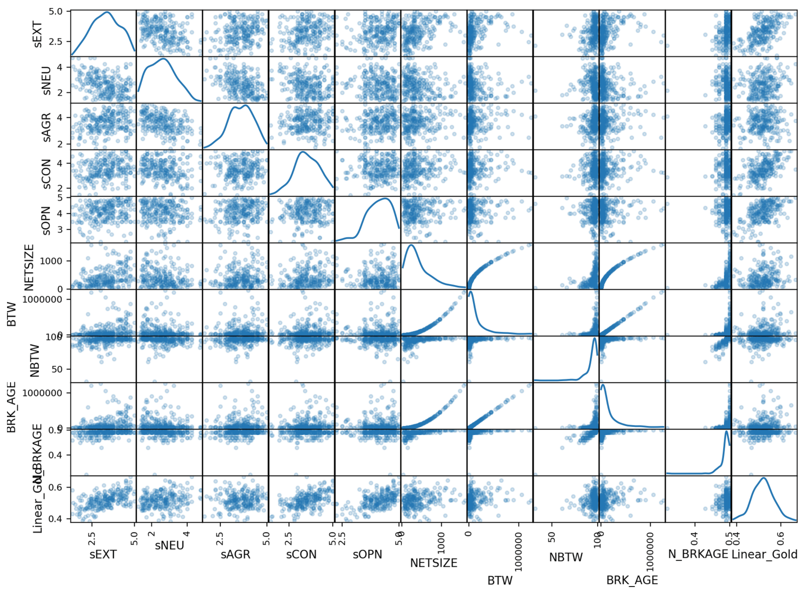


Fig. 3. Correlation between different features in the dataset excluding Facebook status updates. The features here are network size (NETSIZE), betweenness (BTW), the number of betweenness (NBTW), density (DEN), brokerage (BRK_AGE), the number of brokerage (NBRKAGE), and *Linear_ Gold* is the implicit gold regression values.

4 Experimental Methodology

Dataset. We evaluate our methods on myPersonality data[2]. It contains personality scores and Facebook profile data, collected by Stillwell and Kosinski by means of a Facebook application that implements the FFM personality questionnaire in a 100-item long version of [24]. The application obtained the consent from its users to record their data and use it for the research purposes. They selected only the users for which they had both information about personality and social network structure. The status updates have been manually anonymized. The final dataset contains 9,917 Facebook statuses of 250 users in raw text, gold standard (self-assessed) personality labels, and several social network measures. Figure 3 shows the relations between different features in the data. As we can see a higher correlation between openness personal trait to others. Moreover, the data distribution of sOPN and sNEU in the dataset are more imbalanced than other personal traits.

4.1 Gold Standard Values

Ideally, we would evaluate downstream performance compared to a ground truth. Unfortunately, a ground truth is difficult to characterize for the privacy-concern

[2] http://myPersonality.org.

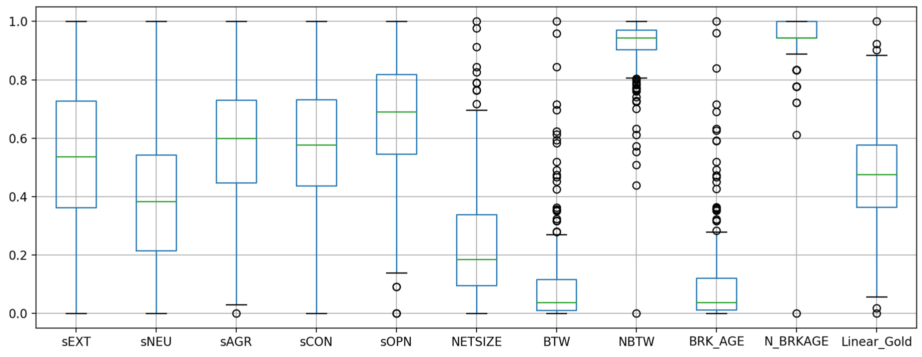


Fig. 4. The distribution of numerical features using box plots. All numbers are normalized to stay between [0, 1] in order to have a good overview of the whole features.

task since people would have answered "as high as possible" if someone simply asked them "how much privacy-guarantee do you want to have?". Our future work would be connecting computer science, crowdsourcing and psychology in order to collect gold standards on user privacy concern using psychological and behavior tests. Since this type of ground truth does not exist currently, we constructed the gold standards in our work as follows, using all available information about users to be an approximation of the ground truth.

Gold Regression Values. The myPersonality dataset contains both personality trait labels and personality trait scores. Therefore, previous works in personality profiling can be catergorized as classification problems [25] or regression problems [26,27]. Since there is no gold privacy-concern labels/values in the dataset, we implicitly derive the values using a linear combination between personality trait scores and the weighting vector V adopt from [18]. We denote $P = (r_i \mid i = \{1, 2 \ldots N\})$, where N is the number of samples and r is the real privacy concern degree of a certain user. Then, $r = \frac{1}{125} * \sum_{i=1}^{5} v_i * s_j$, where $V = (v_i \mid i = 1, 2 \ldots, 5) = (5.0, 4.7, 4.3, 4.1, 1.0)$ taken from [18], and 125 is a normalisation constant. $S = (s_j \mid j = 1, 2 \ldots, 5)$ is user personality trait scores. P is also denoted as *Linear_ Gold* in both Fig. 3 and 4 (i.e., the last column).

Gold Classification Labels. We constructs gold labels to evaluate downstream classification performance followed the work of Vu et al. [9]. The labels are high (HiPC), medium (MePC), and low privacy-concern (LoPC) level as firstly proposed in [16]. Given the ground truth of personality labels, i.e., **yes** and **no** labels of {NEU, OPN, CON, AGR, EXT} - the five personality traits. Based on the findings of [18] we know that privacy concerns of different personality traits are ordered as following {NEU, OPN, CON, AGR, EXT} from the highest privacy-concern to the lowest privacy-concern correspondingly. Thus, we derive the privacy concern levels as in Table 2. Eventually, our ground truth set consists of 29 users in HiPC, 212 users in MePC, and 9 users in LoPC.

Table 2. Deriving self ground-truth labels for classification problem. It is note that MePC are the rest, i.e., {¬HiPC, ¬LoPC}.

Privacy concern label	cNEU	cOPN	cCON	cAGR	cEXT
HiPC	**Yes**	**Yes**	Any	No	No
LoPC	No	No	Any	**Yes**	**Yes**

4.2 Feature Extraction

Feature extraction is a process of extracting valuable and significant information from the raw data to represent the data. Since collecting personally sensitive data is cautious and challenging, the myPersonality dataset is small [28], so we have to incorporate with pre-trained embedding models (e.g., Word2Vec[3]) to better represent the data. Table 3 lists all extracted features in this work inspired by Vu et al. [29]:

Table 3. Total number of extracted features in this work

Feature	# Features	Information
Lexical features	7111	N-grams features, i.e., [1,2,3,4,5]-grams
Topic features	200 & 50	For LSI and LDA features respectively
Semantic features	300	Word2Vec model trained on Google News
Total	7661	

4.3 Evaluation Results

We design two different types of experiments to evaluate our methodology. Firstly, we evaluate how well we can detect privacy-concern regarding both classification problems and regression problems. Secondly, we analyze the effect of personality-based privacy controller to see if our self-adaptive approach can better balance data utility and privacy preservation. Abbreviations used in this section are listed in Table 4.

Privacy-Concern Detection. Using the above ground truth data, similar to [9], we build two different privacy-classifiers with Naive Bayes and Support Vector Machine (SVM) algorithms for the classification task. Table 5 shows the performance of privacy-concern detection in comparison with the work of Vu et al. [9]. In their work, the authors showed that due to the imbalance of class distribution, Naive Bayes (NB) does not perform well. In this work, we solve the imbalanced data distribution using [30]. Table 6 shows the data distribution before and after the over-sampling process. Thus, NB and SVM both perform much better than the majority accuracy.

[3] https://code.google.com/archive/p/word2vec/.

Table 4. List of abbreviation used in this section

#	Abbreviation	Description
1	RMSE	Root Mean Square Error, RMSE$(y, \hat{y}) = \sqrt{\frac{\sum_{t=1}^{n} (\hat{y}_t - y_t)^2}{n}}$
2	EVS	Explained Variance Score, EVS$(y, \hat{y}) = 1 - \frac{Var\{y - \hat{y}\}}{Var\{y\}}$
3	OOBudget	Out of budget records saying number (ratio) of records that get out of budget at a certain iteration
4	MLP-OOBudget	OOBudget of the multi-layer perceptrons algorithm
5	SVR-OOBudget	OOBudget of the Support Vector Regression algorithm
6	LR-OOBudget	OOBudget of the Linear Regression algorithm

Table 7 shows evaluation results of the regression-based problem with three different algorithms including: LR (Linear regression [31]), SVR (SVM regression [31]), and MLP (our proposed multi-layer perception neural network model with a linear layer). The data was divided to 80% for training and 20% for testing. Clearly, LR performs well on the training process, however, it was over-fitted the data (i.e., RMSE = 0, variance score = 1) and could not generalize well to predict testing data. In contrast, MLP performs reasonably well on the training data and achieves the best performance on the test data. It is worth to mention that personality trait scores were used to derive the *Linear_ Gold* values (i.e., P), therefore, they are not included in the feature extraction process. This approach is also closer to the real scenario where we can easily collect UGC but not their personality scores.

Table 5. Privacy concern detection performance in comparison with majority accuracy. The evaluation accuracy of this work is the average accuracy of 5-fold cross-validation.

Paper	Vu et al. [9]			This work		
Algorithm	Majority	Naive Bayes	SVM	Majority	Naive Bayes	SVM
Accuracy	0.78	0.57	0.80	0.848	**0.97**	0.967

Table 6. Label distribution before and after over-sampling using SMOTE [30] to solve the imbalanced data distribution issue.

	Before over-sampling			After over-sampling		
Labels	LoPC	MePC	HiPC	LoPC	MePC	HiPC
# of samples	9	212	29	212	212	212

Table 7. Evaluation results of regression-based privacy concern detection. Note that the higher values of EVS are, the better (with the best possible score of 1.0).

	Algorithm	RMSE	EVS
Training	LR	0	1
	SVR	4.460	−8834
	MLP	0.058	−0.472
Testing	LR	0.064	−0.602
	SVR	6.277	−15252
	MLP	**0.052**	**−0.305**

Privacy-Budget Controller. We design a learning task using SVM with the privacy-budget controller to see how it affects the classification performance. A 10-fold SVM classification is designed to interactively request valid user records until it receives no records. Thus, this test is similar to a real scenario where an analyst requests to the system and retrieves information. Figure 5 shows four different privacy budget controls including (a) global privacy budget, (b) random privacy budget, (c) linear regression based privacy budget, and (d) MLP privacy based budget.

Based on the experimental results, we have the following observations:

(1) Data utility of the global privacy budget quickly drops to 0 due to all records run out of privacy budget at the same time.
(2) Except of global privacy budget, the other three personalized-privacy budgets have better trade-off of privacy and data utility since they can avoid a situation where all records run out of privacy-budget at the same time.
(3) The random privacy-budget achieves better results in terms of data utility, however, it does not take into account the user privacy-concern level.
(4) MLP privacy-budget certainly shows a better way of controlling privacy and data utility. The privacy-budget gradually increases which allows data analysts, i.e., a classification algorithm in our experiment, to receive enough data records to maintain good classification results.

To the regression problem, Fig. 6 shows evaluation results on 50 instances of the testing data. Our proposed MLP-budget controller clearly works better than others in terms of the following criteria:

(1) Regarding performance: we consider the Gold-RMSE (see Fig. 6-(a)) is the standard. Comparing to the Gold-RMSE, MLP-RMSE and SVR-RMSE have the same trend. However, the mean distance of MLP-RMSE to the Gold-RMSE is 9.487, which is smaller than that of SVR-RMSE, i.e., 11.113 (see Table 8).
(2) Regarding OOBudget: LR-OOBudget is much similar to Gold-OOBudget comparing to SVR-OOBudget and MLP-OOBudget. However, its LR-RMSE was the worst.

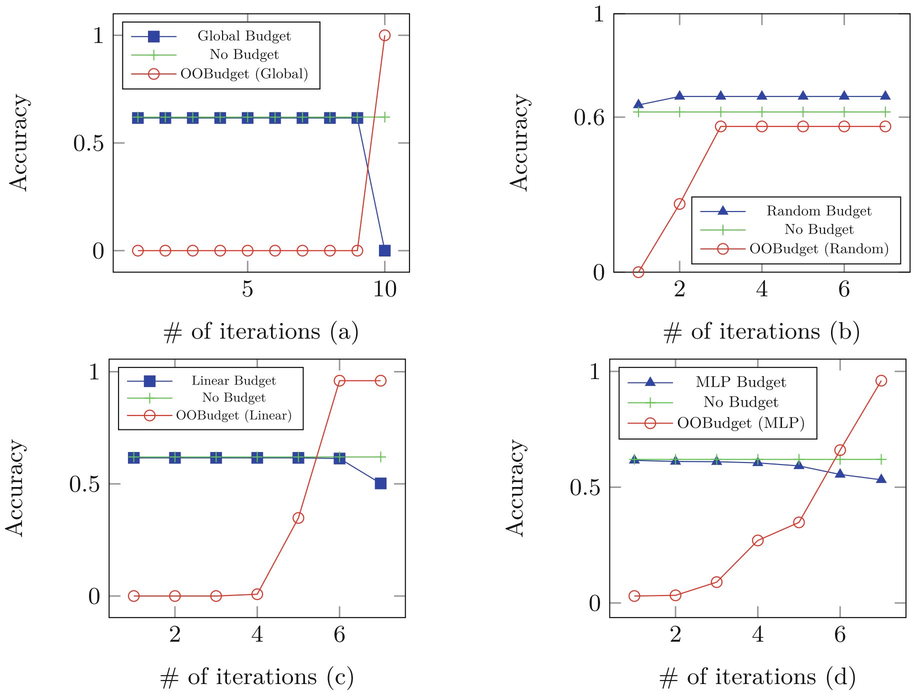


Fig. 5. Evaluating the effect of different privacy-budget control methods to the binary classification performance of the cEXT class. OOBudget is the ratio of out-of-budget user records.

Table 8. Distance to the gold regression in comparison to LR, SVR, and MLP. The iteration running from 1 to 30, but we only show from the iteration 23rd where we can observe OOBudget.

Algorithm and Iteration	23	24	25	26	27	28	29	30	Distance (RMSE)
Gold	1	3	4	14	21	28	35	42	0
LR	0	1	4	7	13	26	38	45	4.183
SVR	0	0	0	0	1	39	49	49	9.487
MLP	0	0	0	0	4	26	48	48	11.113

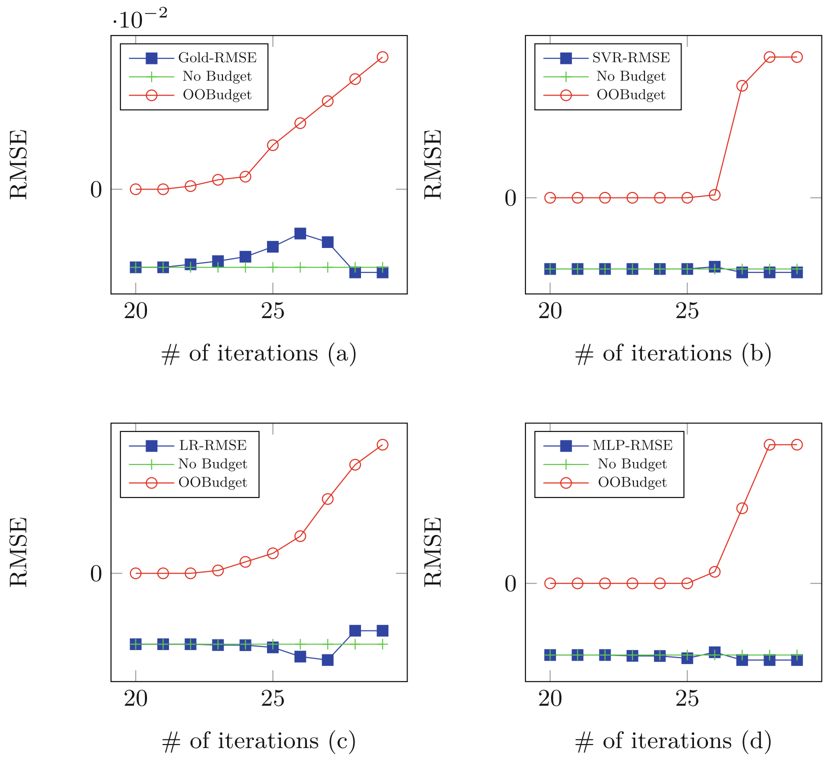


Fig. 6. Evaluating the effect of different privacy-budget control methods to the regression problem performance of the testing data. OOBudget is the ratio of out-of-budget user records.

5 Conclusions and Future Work

This paper presents a self-adaptive differential privacy preserving approach for data analysis. To address the limitation of unified privacy budget in differential privacy, we calculate privacy budget based on personality knowledge. According to our experiments, personality-based privacy budget shows a more practical way of controlling privacy and data utility. Moreover, our proposed approach (i.e., MLP privacy budget) shows the best trade-off of data utility and privacy control. Our approach is applicable to real scenario where we only have user-generated content information. As mentioned before, there is no gold standard values (or labels) from users regarding their privacy-concern. Therefore, one of our future work directions is applying crowdsourcing technologies in user privacy concern detection to contribute the construction of a ground truth framework.

References

1. Papernot, N., McDaniel, P.D., Sinha, A., Wellman, M.P.: Towards the science of security and privacy in machine learning. CoRR (2016)
2. McKenzie, P.J., Burkell, J., Wong, L., Whippey, C., Trosow, S.E., McNally, M.B.: User-generated online content 1: overview, current state and context. First Monday **17**, 4–6 (2012)
3. Narayanan, A., Shmatikov, V.: Robust de-anonymization of large sparse datasets. In: Proceedings of the 2008 IEEE Symposium on Security and Privacy, pp. 111–125. SP 2008 (2008)
4. Fredrikson, M., Jha, S., Ristenpart, T.: Model inversion attacks that exploit confidence information and basic countermeasures. In: Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security, pp. 1322–1333. CCS 2015 (2015)
5. Pennebaker, J.W., King, L.A.: Linguistic styles: language use as an individual difference. J. Personal. Soc. Psychol. **77**, 1296–1312 (1999)
6. Flekova, L., Gurevych, I.: Personality profiling of fictional characters using sense-level links between lexical resources. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1805–1816 (2015)
7. Bollen, J., Mao, H., Zeng, X.: Twitter mood predicts the stock market. CoRR (2010)
8. Flekova, L., Gurevych, I.: Can we hide in the web? Large scale simultaneous age and gender author profiling in social media notebook for PAN at CLEF 2013. In: Working Notes for CLEF 2013 Conference, Valencia, Spain, 23–26 September 2013 (2013)
9. Vu, X.S., Jiang, L., Brändström, A., Elmroth, E.: Personality-based knowledge extraction for privacy-preserving data analysis. In: Proceedings of the Knowledge Capture Conference, pp. 45:1–45:4. K-CAP 2017 (2017)
10. Cynthia, D.: Differential privacy. In: ICALP, pp. 1–12 (2006) (2006)
11. Mairesse, F., Walker, M.A., Mehl, M.R., Moore, R.K.: Using linguistic cues for the automatic recognition of personality in conversation and text. J. Artif. Int. Res. **30**, 457–500 (2007)
12. Bayardo, R.J., Agrawal, R.: Data privacy through optimal k-anonymization. In: ICDE, pp. 217–228 (2005)
13. Wang, R., Wang, X., Li, Z., Tang, H., Reiter, M.K., Dong, Z.: Privacy-preserving genomic computation through program specialization. In: CCS , pp. 338–347(2009)
14. McSherry, F.D.: Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In: SIGMOD (2009)
15. Mohan, P., Thakurta, A., Shi, E., Song, D., Culler, D.: GUPT: privacy preserving data analysis made easy. In: SIGMOD (2012)
16. Ebadi, H., Sands, D., Schneider, G.: Differential privacy: now it's getting personal. In: Proceedings of the 42Nd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages. POPL 2015, pp. 69–81 (2015)
17. Jorgensen, Z., Yu, T., Cormode, G.: Conservative or liberal? Personalized differential privacy. In: 2015 IEEE 31st International Conference on Data Engineering, pp. 1023–1034 (2015)
18. Sumner, C., Byers, A., Shearing, M.: Determining personality traits and privacy concerns from Facebook activity. In: Black Hat Briefings, pp. 197–221 (2011)

19. John, O.P., Srivastava, S.: The big five trait taxonomy: History, measurement, and theoretical perspectives. In: Handbook of Personality: Theory and Research, pp. 102–138 (1999)
20. Murtagh, F.: Multilayer perceptrons for classification and regression. Neurocomputing **2**(5), 183–197 (1991)
21. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. CoRR abs/1412.6980 (2014)
22. Rumelhart, D.E., Durbin, R., Golden, R., Chauvin, Y.: Backpropagation, pp. 1–34 (1995)
23. Abadi, M., et al.: TensorFlow: Large-scale machine learning on heterogeneous systems (2015). https://www.tensorflow.org/
24. Costa, P.T., McCrae, R.R.: The Revised NEO Personality Inventory (NEO-PI-R), pp. 179–198 (2008)
25. Majumder, N., Poria, S., Gelbukh, A., Cambria, E.: Deep learning-based document modeling for personality detection from text. IEEE Intell. Syst. **32**, 74–79 (2017)
26. Farnadi, G., Zoghbi, S., Moens, M.: Cock. Recognising personality traits using Facebook status updates, M.D., pp. 14–18 (2013)
27. Farnadi, G., et al.: Computational personality recognition in social media. User Model. User-Adapt. Interact. **26**, 109–142 (2016)
28. Vu, X.S., Flekova, L., Jiang, L., Gurevych, I.: Lexical-semantic resources: yet powerful resources for automatic personality classification. In: Proceedings of the 9th Global WordNet Conference (2018)
29. Vu, T., Nguyen, D.Q., Vu, X.S., Nguyen, D.Q., Trenell, M.: Nihrio at semeval-2018 task 3: a simple and accurate neuralnetwork model for irony detection in twitter. In: Proceedings of the 12nd International Workshop on Semantic Evaluation (SemEval-2018), pp. 525–530. Association for Computational Linguistics (2018)
30. Bowyer, K.W., Chawla, N.V., Hall, L.O., Kegelmeyer, W.P.: SMOTE: synthetic minority over-sampling technique. J. Artif. Intell. Res. **16**, 321–357 (2011)
31. Pedregosa, F., et al.: Scikit-learn: machine learning in Python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)

Complexity of Russian Academic Texts as the Function of Syntactic Parameters

Valery Solovyev[1](✉), Marina Solnyshkina[1], Vladimir Ivanov[2], and Svetlana Timoshenko[3]

[1] Kazan Federal University, Kazan, Russia
maki.solovyev@mail.ru
[2] Innopolis University, Innopolis, Russia
[3] Institute for Information Transmission Problems, Moscow, Russia

Abstract. Providing students with academic materials of steadily increasing complexity is equally critical for textbook publishers, educators, and test developers, therefore automation of text complexity assessment is relevant in a number of areas. The authors propose an innovative approach addressing limitations of the existing Russian text complexity tools based on descriptive and lexical features only. We examined the impact of 75 syntactic features on texts measured with ETAP-3, contrasted them in texts of different grade levels in Russian Readability Corpus of 1.2 million tokens and suggest an original model of text complexity. We confirm statistically significant correlation of text complexity and 28 syntactic features in Russian texts. More research is needed to examine the models accuracy in texts of different types and genres as well as its applicability in automated text analyzers.

Keywords: Text complexity · Syntactic features · Russian language

1 Introduction

Complexity and informativeness of texts has a significant impact on readers' comprehension in all spheres of life. Readers', i.e. customers' response is equally important in law, medicine, tourism, mass media, etc. The effect of readers' cognitive and language abilities on text comprehension and motivation is scientifically confirmed by numerous research [1]. Adequate text comprehension implies that texts are aligned to readers' needs and interests, and as such establish a welcoming and inquiry-oriented professional environment. The idea behind the existing practice in schools, for example, is to provide students with age-appropriate learning materials and tasks which are neither too arduous nor too effortless for them. Since age standards are difficult to define and measure, the idea of "age-appropriateness" is viewed as one of the most ambiguous and evasive in complexity studies. The latter is caused by the fact that the same books may be 'appropriate' for readers of different grade levels and age, and as such levelled

https://doi.org/10.1007/978-3-031-23793-5_15

differently across reading programs.[1] Aside from that, different text levelling systems employ unsimilar algorithms and indices to measure age-appropriateness. For example, the algorithm of reader-text matching in Lexile Framework is based on word frequency and sentence length.[2] Guided Reading Level, or DRL, tailors students' reading program based on word complexity and text content[3]. Grade level reading texts descriptors (GLRD) define what students of a certain level are expected to know and what types of texts they are able to read and comprehend[4].

GLRD specify students' competencies and the language of a particular course which students at a certain education stage (or grade) are exposed to. One of the most recognized tools in English speaking countries is 'The F&P Text Level Gradient' used to select books for various groups of readers.[5] UK educators are recommended to select reading texts based on 'Grade descriptors for GCSEs graded 9 to 1'.[6] In addition to qualitative descriptions of recommended text types and readers' competencies, educators and parents have numerous online instruments to measure complexity of a certain text. Modern English text complexity analyzers available online have a 20 year history and are quite reliable.[7,8] The existing automatic servers evaluate up to over a hundred text parameters which range from from quantitative, i.e. word count and sentence length, to semantic indices, i.e.coherence, cohesion, sentiment analysis, etc.[9,10]. Coh-Metrix, for instance, an engine developed by SoLET, assesses 108 textual parameters, including narrativity, syntactic simplicity, word concreteness, referential cohesion and deep cohesion [5]. It also reports on texts grade level based on Flesch-Kincaid Grade Level readability formula (FKGL) for native speakers [6] and RDL2 which is the second language readability score applicable in ESOL teaching.

The situation with measuring reader-text matching in Russia is different. Significant gaps have been reported in the area: textbooks and educational materials are either too easy or too difficult for students.

Researchers also indicate that in the majority of cases lack of willingness or ability to read on the part of students of all educational levels is caused by

[1] https://www.psychologytoday.com/us/blog/reading-minds/201702/three-myths-about-reading-levels.

[2] https://hub.lexile.com/lexile-grade-level-charts.

[3] https://www.scholastic.com/teachers/articles/teaching-content/leveled-reading-systems-explained/.

[4] https://schools.cms.k12.nc.us/corneliusES/Documents/Fountas-Pinnell%20Guided%20Reading%20Text%20Level%20Descriptions.pdf.

[5] https://www.gov.uk/government/publications/grade-descriptors-for-gcses-graded-9-to-1/grade-descriptors-for-gcses-graded-9-to-1-english-language.

[6] https://www.gov.uk/government/publications/grade-descriptors-for-gcses-graded-9-to-1/grade-descriptors-for-gcses-graded-9-to-1-english-language.

[7] http://www.readabilityformulas.com/free-readability-formula-tests.php.

[8] http://readability.io/.

[9] https://readable.io/text/.

[10] http://casemed.case.edu/cpcpold/students/module4/WordReadability.pdf.

inappropriate selection of a book by an educator[11] Unfortunately, Russian text complexity analyzers so far apply no other variables but quantitative, i.e. word length and sentence length [9]. The online available Russian-language texts analyzers, RuLingva (https://rulingva.kpfu.ru/) and Textometr (https://textometr.ru/) identify and report on descriptive (qualitative) measures including morphological (POS classification) and lexical indices. Their models for measuring text complexity lack the capacity to use grammar as an overt component of the algorithm. In the paper we aim at the following research question: What syntactic features relate to complexity of Russian academic texts?

2 Background

The modern paradigm of complexity theory developed in the 20th century is based on the idea that an object's complexity is measured with the amount of information required to describe it [11]. Thus, the method of complexity assessment depends on the object and its inherent characteristics: its integrity, functionality, dynamism, structure, etc. The modern model for determining an object complexity is based on a variety of features inherent to the object and is defined as an additive or non-additive category. Additive or descriptive complexity, i.e. the complexity, reduced to the sum of components and relations between them, is inherent to objects of a simple structure [11]. Dynamic objects exhibit a non-additive complexity [20] Complexity is estimated based on "the amount of information required to resolve (reduce) the uncertainty (ambiguity) of the system" [11]. This paradigm can serve as a basis for assessing the so-called 'descriptive' complexity of a text as a deterministic system, measured on the basis of the complexity of its constituent elements (syllables, words, sentences) [22]. In a number of research, this complexity is referred to as structural, and when being assessed, researchers determine the number of elements, levels, number and variety of connections between elements (hierarchical, functional, causal, spatio-temporal, etc. [11]. Within the frames of this approach traditionally identified as parametric, text complexity is viewed as an objective value and is assessed as the sum of 'complexities of its individual parameters'.

Accepting I. Lerners' point of view V. Tcetlin specified that "complexity of any educational material is its objective characteristics" [21]. Though the latter were specified differently depending on the availability of measures [14] the range always included readability [16]. In modern text complexity models, syntactic complexity is viewed as an explicit and distinct component due to the fact that grammar affects reading comprehension [4]. Syntactic complexity was salient in the text complexity studies in 1970s s and 1980s s [10]. Although, the list of syntactic features viewed by researchers as functions of text complexity traditionally include sentence length, incidence of functional words, number of discourse markers, active voice, multi-word noun phrases, anaphoric references,

[11] https://www.science-education.ru/ru/article/view?id=22229, https://cyberleninka.ru/article/v/chtenie-kak-sotsialnaya-problema, https://trv-science.ru/2015/11/17/pochemu-ne-chitayut-shkolniki/.

inversion, number of clauses, etc. their impact in texts of different languages may vary. The best evidence of the above are readability formulas. The general approach to "text readability" once introduced by M. Vogel and K. Washburn was taken as a foundation in all subsequent works. It is based on the objective characteristics of the text highly correlating with comprehension test results, and implies designing a regression equation between a text comprehension, on the one hand, and text parameters – on the other [12].

The first attempts to assess Russian texts readability were made in late 1970s s [3]. The formulas proposed to predict Russian texts readability are based on a number of objective variables borrowed from similar English texts complexity formulas, i.e. word length and sentence length. Readability measurement was initially performed manually, and only in the 2010s, automatic tools were developed to assess grade levels of Russian texts. They were all based on I. Oborneva's readability formula derived in 2006 [9]:

$$FRE = 206.836 - (1.52 \cdot ASL) - (65,14 \cdot ASW)$$

where ASL is the average sentence length, i.e., the number of words divided by the number of sentences; ASW is the average number of syllables per word, i.e., the number of syllables divided by the number of words in a text. A decade later a new study confirmed that I.V. Oborneva's formula, modelled on fiction, elevates readability grade levels of informational texts [18] and as such can not be employed on all types of Russian texts. The researchers confirmed genre dependency of all readability formulas and proposed a new formula aimed at measuring readability of informative expository texts: FKGL = 0.36 * ASL + 5.76 * ASW - 11.97 [15].

Syntactic parameters impact has been confirmed in a number of Russian text complexity studies. For example, the authors in [8] analyzed the frequency of different parts of speech in texts of different complexity levels: the share of grammatical features was calculated both in the text and in a sentence. The finding indicate that correlations are statistically significant for the frequency of syntactic parameters when measured in a text, not a sentence. In [19], the authors conducted analysis of 24 morphosyntactic features in 14 academic texts. However, neither of these studies examined the impact of 75 syntactic parameters on complexity in 26 classroom books. To the best of our knowledge, the assessment of Russian text complexity has never been modelled as a function of this variety of syntactic parameters.

3 Datasets

We assembled two collections of textbooks for the research: (1) 14 textbooks on Social Studies by L. N. Bogolubov and A.F. Nikitin marked "NIK" ; (2) 12 Biology textbooks which written and edited by different authors. Further we refer to the two collection as a Russian Readability Corpus (RRC). All textbooks are recommended by the Ministry of Education and Science of the Russian Federation to Use in Secondary and High Schools"[12].

To ensure reproducibility of results, we uploaded the corpus on a website thus providing its availability online[13]. Note, however, that the published texts contain shuffled order of sentences. The sizes of the collections of texts are presented in Table 1.

Table 1. Statistics for documents in corpus. Note. Sign ('+') marks the textbooks of advanced levels of complexity.

Train set (13 documents)				Test set (13 documents)			
Textbook/grade	Tokens	ASW	ASL	Textbook/grade	Tokens	ASW	ASL
Biology/ 5	15,555	2.728	10.198	Social Nik/5	17,221	2.231	7.219
Biology/5	17,684	2.573	9.502	Social Nik/6	16,475	2.527	10.342
Biology/5–6	23,689	2.759	11.351	Social Nik/7	22,924	2.525	10.519
Biology/7	51,954	2.693	11.999	Social Nik/8	40,053	2.693	11.256
Biology/9	49,611	2.917	10.807	Social Nik/9	43,404	2.809	11.205
Biology/10–11	62,568	2.968	13.946	Social Nik/10	39,183	2.878	13.686
Social Bog/6	16,467	2.417	10.914	Social Nik/11	38,869	2.877	14.184
Social Bog/7	23,069	2.680	11.536	Biology/5	22,361	2.631	9.327
Social Bog/8	49,796	2.779	14.079	Biology/5–6	42,808	2.630	10.265
Social Bog/9	42,305	2.846	14.71	Biology/7	40,622	2.783	9.646
Social Bog/10	75,182	2.867	15.116	Biology/9	56,503	2.850	13.381
Social Bog/10+	98,034	2.849	15.588	Biology/10–11	66,429	2.866	13.100
Social Bog/11+	100,800	2.982	15.563	Biology/ 10–11	81,712	2.860	13.759

4 Methods

A priori consideration of numerous morphosyntasic features reveals that incidence of some of them is to be measured in a text, while with others are to be calculated as an average in a sentence. Among the first group of features are incidences of different parts-of-speech. For example, the average number of nouns in a sentence will obviously grow along with the length of a sentence and as such the feature is unlikely to shed light on the problem. At the same time, the incidence of nouns in the text is irrespective of the sentence length and can be viewed as a text parameter. On the other hand a number of syntactic features, such as the longest_path, i.e. the average length of the longest branch, reflecting the structural properties of the syntactic tree in a sentence, can obviously be measured in a sentence. Thus, all morphosyntactic features are measured in two different ways: as (1) frequency of occurrence throughout text, (2) frequency of occurrence in a sentence. See Appendix for the complete list of features.

[12] http://www.fpu.edu.ru/fpu/.
[13] https://kpfu.ru/slozhnost-tekstov-304364.html.

4.1 Corpus Preprocessing

All the texts under study were prepossed in the same way which included tokenization, splitting text into sentences and part-of-speech tagging (with Russian TreeTagger[14]). We also deleted all sentences longer than 120 words and shorter than 5 words, both of which are considered outliers in secondary and high school textbooks.

4.2 Syntactic Level Features

We have explored an extended feature set for text complexity modeling. Metrics of each of the selected features were calculated with ETAP-3. "ETAP-3" as a multipurpose linguistic processor enables to extract and measure numerous syntactic features. Its parser models dependency tree structures, the nodes of which are word tokens of the input sentence, and 'the edges' are the established syntactic dependencies. Thus, every tree node corresponds to a word token in the sentence processed, whilst the directed arcs are labeled with names of syntactic relations [2]. All the syntactic dependencies have directions, therefore all the dependencies have an original node, its host, and the final one, i.e. the dependent node. Each word token is represented as a lemma and a set of its morphological characteristics [2]. All the texts in the collection were processed with the syntactic parser: each sentence was converted into a dependency tree structure. The algorithm of metrics assessment is presented in detail in our previous work [19]. We estimated correlation coefficient of each syntactic feature under study and text readability marked as Grade level. (See the results in Table 3 in Appendix).

4.3 Linear Regression Model

We tested the features for their significance in linear regression model. The model was trained on the first half of the texts from (see Table 1) and tested on the second group of texts. Both parts are balanced in terms of the number of Biology and Social Science texts. The quality of the proposed models was assessed with the mean squared error (MSE), mean absolute error (MAE) and determination coefficient R2.

In general, some syntactic feature are similar to others and correlate with the target variable (readability, referred to as a grade level). However, it is evident that all the syntactic features have lower correlation coefficient with the target feature ('Grade Level'), than the 'classical' features ASL and ASW.

Nevertheless, syntactic features have high correlation with the target variable. This information could be useful for readability and text complexity prediction in Russian. Our goal is to evaluate syntactic features in the next subsection with respect to the capability to serve as predictors in a linear regression model. As shown in Table 3 few of the syntactic features have low correlation with the target variable ('Grade Level'). In the rest of the paper we consider only those syntactic features with correlation coefficient above 0.5 or less than -0.5 (as depicted in Table 3).

[14] http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/.

4.4 Evaluation of Syntactic Features

For evaluation of the syntactic features we carried out another experiment. We clustered the syntactic features with respect to their similarity to each other. We view similarity as the correlation of the features' metrics derived in RRC. The derived groups of features are the following:

- Group 1, (G-1): Features related to the structure of the syntax tree
  - leaves_number (LN)
  - average_path (AP)
  - longest_path (LP)
  - path_number (PN)
- Group 2, (G-2): Features related to coordinating constructions
  - average_sochin_length (SL)
  - sochin_number (SN)

The following groups of features are: NOUN_related, PART_related, POS_related, VERB_related, SUB_CLAUSE, syntax, OTHER as they are highlighted in Table 3 in the Type column. Group ALL includes all features, Group BEST contains features with the highest correlation coefficients provided in Table 3: 'ASL', 'nouns_c_acc', 'nouns_c_gen', 'nouns_c_nom', 'part_adj_c_dat', 'part_adj_c_gen', 'part_adj_c_nom', 'part_adj_p_post', 'pos_adj', 'pos_verb', 'predicate_number', 'prich_rate', 'sochin_number', 'sub_clause_t_adj_conc', 'sub_clause_t_adj_goal', 'verbs_f_imp', 'verbs_f_ind', verbs_f_part', 'verbs_n_plur', 'verbs_p_2', 'ASW', 'average_path', 'average_sochin_length', 'deeprich_v', 'leaves_number', 'longest_path', 'path_number', 'prich_v'.

Thus, the selected syntactic features could serve better predictors in a linear regression model due to their high correlation with the target variable. The results are provided in Table 2.

Table 2. Results features evaluation in linear models.

GROUP	MSE	MAE	R^2
Shallow	2.210	1.256	0.847
G-1	12.668	2.737	0.835
G-2	2.520	1.328	0.797
NOUN_related	3.366	1.547	0.984
PART_related	36.622	5.067	0.891
POS_related	10.244	2.972	0.955
SUB_CLAUSE	16.081	3.119	0.948
VERB_related	4.978	1.761	1.000
Syntax	6.501	1.990	0.904
OTHER	4.687	1.698	0.906
ALL	3.556	1.565	1.000
BEST	2.806	1.426	1.000

5 Discussion

In earlier studies [18], the authors investigated text complexity formulas based on two simple parameters only, i.e. ASW and ASL. Our findings were quite conclusive: two features, though quite significant, can not be the only proxies of text complexity. In [15], our focus was on a set of lexical features analyzed with machine learning methods and the algorithm was piloted on RRC with its size of over 200,000 tokens. The accuracy of the model we applied in [15] was significantly higher than that of similar studies of Russian lexical features, however its limitation was obvious: text complexity is not a function of lexical features only. We share the view point of numerous researchers in the area that that syntactic features are powerful variables in text complexity. Therefore, in this research we tackle impact of 75 fine-grained syntactic features on text complexity thus scrutinizing it from multiple perspectives. We take it for granted that text comprehension depends on a set of text parameters in which syntactic features play a critical role.

With the ultimate goal to determine how grade level responds to different features of syntactic complexity we processed all possible syntactic features of 26 textbooks with ETAP-3. Also, the choice of a parser for the Russian language is natural because the parser of ETAP-3 is the most powerful of the existing parsers of the Russian language. Based on experts' assessment we selected 75 features which were supposed to impact text complexity.

This study confirmed the important role of ASL, i.e. the average sentence length, in defining text complexity. The study also endorses the idea of a high correlation of text complexity with the incidence of participles in a text and anti-correlation with the incidence of nouns in the nominative case [8]. The list of Russian text complexity predictors we identified in this study has never been verified before. It includes features defining structural complexity of syntactic dependency trees: path number, leaves number and some others.

Paradoxically, the best results were demonstrated by the largest group, consisting of all features and the smallest, including the two features, i.e. ASW and ASL. The perspective of the research lies in processing the group of features including coordinating constructions: average_sochin_length and sochin_number which we presume may serve as good predictors of syntactic complexity of Russian texts.

6 Conclusion

The article presents the results of the first systematic study of syntactic features as predictors of Russian texts complexity. We extracted, compared and contrasted 75 syntactic features in two sets of 26 textbooks with the total size of 1,138,827 tokens. The findings demonstrate a wide range of correlation coefficients indicating a different degree of influence of syntactic features on text complexity. The most significant parameters identified include the following: average number of subtrees and average number of leaves. The research findings have a potential to be used in machine learning systems, including neural networks. The list of the syntactic parameters most influencing text complexity can be used in text simplification procedures by textbook writers and test developers. In further studies, we plan to employ neural networks with deep learning [7], as well as to further expand the list of parameters, including n-grams [13] and other parameters [17].

Acknowledgements. This paper has been supported by the Strategic Academic Leadership Program of Kazan Federal University (PRIORITY-2030).

Appendix

Table 3. Features and their correlation (ρ) with the 'Grade Level'.

ID	Type	Feature	Description	ρ	p-value
1	shallow	ASL	Average sentence length	0.8738	0.0000
2	syntax	path_number	Average number of subtrees	0.8540	0.0000
3	syntax	leaves_number	Average number of leaves	0.8462	0.0000
4	participle_ related	part_adj_c_dat	The average number of adjectives in the dative case	0.8431	0.0000
5	syntax	longest_path	Average length of the longest branch	0.8365	0.0000
6	syntax	average_path	Average branch length	0.8325	0.0000
7	syntax	deeprich_v	average span of a verbal adverb phrase	0.7601	0.0000
8	syntax	prich_v	the average span of a participial construction	0.7539	0.0000
9	shallow	ASW	average syllables per word	0.7488	0.0000
10	participle_ related	part_adj_p_post	Average number of participial constructions l in postposition	0.7208	0.0000
11	participle_ related	part_adj_c_gen	Average number of participial constructions in the dative case	0.7111	0.0000
12	syntax	average_ sochin_length	the average length of coordinating constructions	0.6592	0.0002
13	other	prich_rate	Average number of participial constructions	0.6206	0.0007
14	POS_related	pos_adj	Average number of adjectives	0.5921	0.0014
15	VERB_related	verbs_f_part	Average number of participles	0.5818	0.0018
16	NOUN_related	nouns_c_gen	Average number of nouns in genitive case	0.5725	0.0022
17	participle_ related	part_adj_c_nom	Average number of participial constructions in the nominative case	0.5060	0.0084
18	SUB_CLAUSE	sub_clause_ t_adj_conc	Average number of concessive clauses	0.5047	0.0086
19	SUB_CLAUSE	sub_clause_t_rel	Average number of relative clauses	0.4521	0.0204
20	participle_ related	part_adj_c_inst	Average number of participial constructions in the instrumental case	0.4319	0.0276
21	participle_ related	part_adj_p_pre	Average number of participle constructions in preposition	0.4167	0.0342
22	NOUN_related	nouns_g_neu	Average number of neuter nouns	0.3908	0.0484
23	participle_ related	part_adj_c_prep	Average number of participial constructions in the prepositional case	0.3687	0.0638
24	NOUN_related	nouns_g_fem	Average number of feminine nouns	0.3627	0.0686
25	VERB_related	verbs_g_neu	Average number of neuter verbs	0.3575	0.0730
26	participle_ related	part_adj_c_acc	Average number of participial constructions in the accusative case	0.3479	0.0816
27	NOUN_related	nouns_n_sing	Average number of singular nouns	0.3260	0.1041
28	VERB_related	verbs_g_fem	Average number of feminine verbs	0.2449	0.2278

(*continued*)

Table 3. (*continued*)

ID	Type	Feature	Description	ρ	p-value
29	NOUN_related	nouns_c_prep	Average number of nouns in the prepositional case	0.2084	0.3070
30	NOUN_related	nouns_c_inst	Average number of nouns in the instrumental case	0.1915	0.3486
31	POS_related	pos_num	Average number of numerals	0.1682	0.4114
32	NOUN_related	nouns_c_dat	Average number of nouns in the dative case	0.1425	0.4873
33	POS_related	pos_noun	Average number of nouns	0.1174	0.5677
34	SUB_CLAUSE	sub_clause_ t_adj_cause	Average number of causal clauses	0.1045	0.6113
35	SUB_CLAUSE	sub_clause_ t_adj_comp	Average number of comparative adverbial clauses	−0.0425	0.8369
36	other	podchin_rate	Average number of subordinations	−0.1350	0.5107
37	VERB_related	verbs_n_sing	Average number of singular verbs	−0.1609	0.4324
38	VERB_related	verbs_t_past	Average number of verbs in the past tense	−0.1845	0.3670
39	SUB_CLAUSE	sub_clause_ t_adj_cond	Average number of conditional clauses	−0.1926	0.3458
40	SUB_CLAUSE	sub_clause_ t_com	Average number of complement clauses	−0.2081	0.3076
41	participle_ related	part_adv_p_pre	Average number of adverbial participial phrases in preposition	−0.2320	0.2541
42	VERB_related	verbs_g_masc	Average number of masculine verbs	−0.2361	0.2456
43	other	podchin_number	Average number of sentences with subordination	−0.2502	0.2178
44	NOUN_related	nouns_n_plur	Average number of plural nouns	−0.2672	0.1869
45	POS_related	pos_conj	Average number of conjunctions	−0.2886	0.1527
46	VERB_related	verbs_f_inf	Average number of verbs in indefinite form	−0.2930	0.1463
47	POS_related	pos_intj	Average number of interjections	−0.3000	0.1364
48	POS_related	pos_prep	Average number of prepositions	−0.3168	0.1148
49	POS_related	pos_part	Average number of particles	−0.3176	0.1139
50	VERB_related	verbs_f_subj	Average number of subjunctive verbs	−0.3240	0.1064
51	other	deeprich_rate	Average number of adverbial participial phrases	−0.3310	0.0986
52	participle_ related	part_adv_p_post	Average number of adverbial participial phrases in postposition	−0.3350	0.0944
53	SUB_CLAUSE	sub_clause_ t_adj_place	Average number of subordinate clause of place	−0.3367	0.0926
54	other	sentsoch_number	Average number of sentimental-compositional connections	−0.3600	0.0708
55	VERB_related	verbs_ t_pres	Average number of verbs in the present tense	−0.3756	0.0586
56	SUB_CLAUSE	sub_clause_ t_adj_manner	Average number of adverbial participial phrases in clauses	−0.3973	0.0445
57	SUB_CLAUSE	sub_clause_ t_adj_time	Average number of adverbial clauses of time	−0.4071	0.0390
58	SUB_CLAUSE	sub_clause_ t_adj	Average number of adverbial clauses, total	−0.4321	0.0275
59	VERB_related	verbs_f_adv	Average number of adverbial participles	−0.4337	0.0269
60	VERB_related	verbs_p_3	Average number of third person verbs	−0.4575	0.0188
61	VERB_related	verbs_p_1	Average number of first person verbs	−0.4698	0.0154
62	other	root_offset	Average distance from the beginning of a sentence to the predicate	−0.4733	0.0146
63	VERB_related	verbs_t_fut	Average number of future tense verbs	−0.4806	0.0129
64	NOUN_related	nouns_g_masc	Average number of masculine nouns	−0.4865	0.0117
65	POS_related	pos_adv	Average number of adverbs	−0.4884	0.0113
66	SUB_CLAUSE	sub_clause_ t_adj_goal	Average number of clauses of purpose	−0.5576	0.0031
67	NOUN_related	nouns_c_nom	Average number of nouns in the nominative case	−0.5919	0.0014
68	VERB_related	verbs_n_plur	Average number of plural verbs	−0.5957	0.0013
69	NOUN_related	nouns_c_acc	Average number of nouns in accusative case	−0.6397	0.0004
70	VERB_related	verbs_p_2	Average number of second person verbs	−0.6874	0.0001
71	VERB_related	verbs_f_imp	Average number of imperative verbs	−0.6966	0.0001
72	POS_related	pos_verb	Average number of verbs	−0.6978	0.0001
73	other	predicate_ number	Average number of predicates	−0.7073	0.0001
74	VERB_related	verbs_f_ind	Average number of verbs in the indicative mood	−0.7436	0.0000
75	other	sochin_number	Average number of coordinating chains	−0.7588	0.0000

References

1. Alexander, P.A., Jetton, T.L.: The role of importance and interest in the processing of text. Educ. Psychol. Rev. **8**(1), 89–121 (1996)
2. Boguslavsky, I.M., et al.: Interactive resolution of intrinsic and translational ambiguity in a machine translation system. In: Gelbukh, A. (ed.) CICLing 2005. LNCS, vol. 3406, pp. 388–399. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-30586-6_42
3. DuBay, W.H.: The principles of readability. Online Submission (2004)
4. Frantz, R.S., Starr, L., Bailey, A.L.: Syntactic complexity as an aspect of text complexity. Educ. Res. **44**(7), 387–393 (2015)
5. Graesser, A.C., et al.: COH-metrix measures text characteristics at multiple levels of language and discourse. Elem. School J. **115**(2), 210–229 (2014)
6. Kincaid, J., Fishburne, R., Jr., Rogers, R., Chissom, B.: Derivation of new readability formulas (automated readability index, fog count and Felsch reading ease formula) for navy enlisted personnel. Technical report, Naval Technical Training Command Millington TN Research Branch (1975)
7. Kulkarni, A., Shivananda, A.: Deep learning for NLP. In: Natural Language Processing Recipes, pp. 213–262. Apress, Berkeley, CA (2021). https://doi.org/10.1007/978-1-4842-7351-7_6
8. Laposhina, N., Veselovskaya, V., Lebedeva, M.U., Kupreshchenko, O.F.: Automated text readability assessment for Russian second language learners. In: Komp'juternaja Lingvistika i Intellektual'nye Tehnologii, pp. 403–413 (2018)
9. Obobroneva, I.: Avtomatizirovannaya otsenka slozhnosti uchebnykh tekstov na osnove statisticheskikh parametrov [semiautomatic evaluation of the complexity of academic texts on the base of statistic parameters]. Moscow: Rae institute of content and methods of teaching. M.: RAS Institut soderzhaniya i metodov obucheniya (2006)
10. Pearson, P.D., Camperell, K.: Comprehension of text structures. Compr. Teach. Res. Rev. **1981**, 27–55 (1981)
11. Romanov, V.N.: Tehnika analiza slozhnyh system [technique of analysis of complex systems]. SZTU Publ, SPb (2011)
12. Shpakovskiy, Y., et al.: Otsenka trudnosti vospriyatiya i optimizatsiya slozhnosti uchebnogo teksta [Evaluation of the difficulty of perception and optimization of the text complexity]. PhD thesis (2007)
13. Sidorov, G., Velasquez, F., Stamatatos, E., Gelbukh, A., Chanona-Hernández, L.: Syntactic dependency-based n-grams as classification features. In: Batyrshin, I., Mendoza, M.G. (eds.) MICAI 2012. LNCS (LNAI), vol. 7630, pp. 1–11. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-37798-3_1
14. Solnyshkina, M., Kiselnikov, A.: Slozhnost'teksta: Etapy izucheniya v otechestvennom prikladnom yazykoznanii [text complexity: Study phases in russian linguistics]. Vestnik Tomskogo gosudarstvennogo universiteta. Filologiya [Tomsk State University J. Philol.]. **6**, 38 (2015)
15. Solnyshkina, M., Ivanov, V., Solovyev, V.: Readability formula for Russian texts: a modified version. In: Batyrshin, I., Martínez-Villaseñor, M.L., Ponce Espinosa, H.E. (eds.) MICAI 2018. LNCS (LNAI), vol. 11289, pp. 132–145. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-04497-8_11
16. Solnyshkina, M.I., Harkova, E.V., Kazachkova, M.B.: The structure of cross-linguistic differences: Meaning and context of 'readability' and its Russian equivalent 'chitabelnost' (2020)

17. Solovyev, V., Ivanov, V.: Knowledge-driven event extraction in Russian: corpus-based linguistic resources. Comput. Intell. Neurosci. **2016**, 16 (2016)
18. Solovyev, V., Ivanov, V., Solnyshkina, M.: Assessment of reading difficulty levels in Russian academic texts: approaches and metrics. J. Intell. Fuzzy Syst. **34**(5), 3049–3058 (2018)
19. Solovyev, V., Solnyshkina, M., Ivanov, V., Rygaev, I.: Computing syntactic parameters for automated text complexity assessment. In: CEUR Workshop Proceedings, pp. 62–71 (2019)
20. Svyatogor, L.A., Gladun, V.P.: Mashinnoe ponimanie tekstov estestvennogo yazyka: ontologicheskaya paradigma. SHtuchnij ntelekt (2010)
21. Tsetlin, V.S.: Didactic requirements: the case of academic texts (didakticheskie trebovaniya k kriteriyam slozhnosti uchebnogo materiala). Novye issledovaniya v pedagogicheskih naukah.-M.: Pedagogika. **1**, 30–33 (1980)
22. Vereshchagin, N., Uspensky, V.A., Shen, A.: Kolmogorov complexity and algorithmic randomness (2013)

An Experimental Approach for Information Extraction in Multi-party Dialogue Discourse

Pegah Alizadeh[1], Peggy Cellier[2], Thierry Charnois[3], Bruno Crémilleux[1](✉), and Albrecht Zimmermann[1]

[1] UNICAEN, ENSICAEN, CNRS - UMR GREYC, Normandie Univ, 14000 Caen, France
{pegah.alizadeh,bruno.cremilleux,albrecht.zimmermann}@unicaen.fr
[2] Univ Rennes, Inria, INSA, CNRS, IRISA, Campus de Beaulieu, 35042 Rennes, France
peggy.cellier@irisa.fr
[3] CNRS-LIPN-Université Sorbonne Paris Nord, Villetaneuse, France
thierry.charnois@lipn.univ-paris13.fr

Abstract. In this paper, we address the task of information extraction for transcript meetings. Meeting documents are not usually well structured and are lacking formatting and punctuations. In addition, the information are distributed over multiple sentences. We experimentally investigate the usefulness of numerical statistics and topic modelling methods on a real dataset containing multi-part dialogue texts. Such information extraction can be used for different tasks, of which we consider two: contrasting *thematically related but distinct* meetings from each other, and contrasting meetings involving *the same participants* from those involving other. In addition to demonstrating the difference between counting and topic modeling results, we also evaluate our experiments with respect to the gold standards provided for the dataset.

Keywords: Information extraction · Dialogue texts · Topic modeling · Term weighting

1 Introduction

Many organizations (and people) spend large amount of their professional time on (or in) meetings. When the meetings are finished, they should be analyzed w.r.t. different aspects such as preparing meeting summaries, lists of envisioned projects, decisions, problems, action points etc. Preparing an automated approach for analyzing meetings would therefore be expected to be a boon for both meeting participants and non-meeting actors, e.g. managers, auditors etc. For the sake of capturing as rich a data set as possible, meetings can be recorded and stored in audio and/or video form. Thanks to new technologies such as Speech-to-Text[1] all registered dialogue during the meetings can be transformed into

[1] Tool example: http://www.vocapia.com.

textual transcripts. In general, speech recognition or natural language processing on dialogue based conversations is difficult because they consist of multi-part interactions with extreme variability. On the other hand, the transcript of meetings – *output* of speech recognition methods – usually includes unstructured word streams, weak punctuation, formatting or capitalizations.

In this paper we experimentally evaluate how to extract information and topics from a meeting based corpus, namely AMI [1,2] (See Sect. 3). We consider this kind of information as the first stepping stone towards summarizing a meeting. It is not obvious which approaches are well-suited to this question but the research literature on extracting representative terms is vast. We chose to evaluate representatives of two different paradigms: a counting statistic, *tf-idf* often used in information retrieval and event detection, and a topic modeling approach, *NMF*.

Term extraction from meetings is obviously not a means to itself but a stepping stone towards more complex tasks. The questions we ask are therefore:

1. Can we identify the terms specific for an individual meeting in comparison to *thematically related but distinct* meetings?
2. Can we identify which meetings should be grouped together, e.g. by identifying common names of participants?
3. To what degree do extracted sets of words agree with the provided gold-standard summarizations?

We contribute to answer these questions by showing that tf-idf approaches perform well on some tasks (e.g. meeting characterisations and speakers identification) within a meeting acted out by a single group speaking about the same scenario. Whereas topic modeling is better for summarization and theme extraction within a set of thematically related meetings acted out by different groups of participants. In the other words, the two approaches complement each other.

In the following section, we give an overview of the related work, and in Sect. 3 we describe the corpus used in the paper. We describe and define the approaches we have evaluated in Sect. 4. Section 5 describes the experimental evaluation and discusses the results for the three settings described above in detail. In the last section, we conclude and outline perspectives.

2 Background and Related Work

In this paper we are interested in spoken multi-party human meetings and our goal is to extract information such as summary, important events, decisions taken during the meeting, identified problems, proposed solutions and decided action points, etc. We are also interested in knowing which topics where discussed in a particular meeting as well as having making the discussion of particular topics accessible.

A meeting analyzer system called CALO has been proposed in [18,19] that first transcribes meeting speech to text automatically, and then analyses the meeting in several phases: dialogue act segmentation and tagging, topic segmentation and identification, problem and decision detection, and summarization.

With respect to this paper's scope, they identify topic and segment meetings using a generative topic model derived from Latent Dirichlet Allocation (LDA) method to learn the topic model automatically [13]. Using an unsupervised learning approach, they produce the meeting segmentations simultaneously to learn *what* (meeting topic) and *when* (meeting segmentation) people talk about during meetings. An alternative approach for segmenting meetings consists of tracking changes in lexical distributions: [5] and [6] translate the lexical distributions into a discriminative classifier and apply it on meeting transcripts.

Action items and decision extraction from meetings is another concern of this paper. A structural approach for detecting items and decisions has been proposed in [18,19]. They classify meeting utterances w.r.t their roles in the process: task definition, agreement, and acceptance of responsibility. Next, they detect actions [12] and decisions [4] from the role patterns. Another approach in the literature extracts important words (related to problems) using classifiers or sequence models using a lexical approach [3].

Tur *et al.* [18,19] extract different shortened version of meetings according to different evaluation measures. They introduce a method that computes *oracles* of summaries and selects the one with maximum performance according to "ROUGE" [16].

An alternative view consists of considering concepts such as decisions, problem identifications, and dialogue acts as *events* occurring during a meeting. Event detection is a well-established [9] and active [7] research field, which in recent years has focused on social media platforms, particularly Twitter [8,10,14,17,20]. Compared to meeting dialogue, Twitter is an interesting case because it shares characteristics with it, such as weak punctuation or non-standard use of words. Topic modeling has been used for event detection [10,14]. Yet other approaches focus on statistical properties of the terms that occur in different documents, identifying"bursty"terms [8,20], i.e. terms that during a given period occur much more often than before or after, and/or clustering them [17] to identify coherent topics.

3 AMI in Focus

The AMI meeting corpus contains 100 meeting hours captured using many synchronised recording devices[2]. All meeting participants, both native or non-native speakers, speak in English. In total, there are 171 meetings divided into two groups: **scenario-based meeting** and **non-scenario based meeting**. A scenario-based meeting simulates the discussions of team of four participants that are developing a remote control from start to prototype. For each simulation, the design process is divided into four parts which are held during a single day [2]. The"non-scenario based"meetings include real recorded meetings about designing various systems, as well as a small subset of scenarios that are different from the remote control one.

[2] Available here: http://groups.inf.ed.ac.uk/ami/download/.

The AMI documents have been transcribed orthographically correct (while maintaining contractions) with annotated subsets of meetings including name entities, dialogues acts, abstractive and extractive summaries [2]. Abstractive summaries are texts of about 200 words for each meeting, consisting of free text giving a general abstract of the meeting plus specific explanations of the decisions, problems or issues encountered during the meeting. Extractive ones identify the parts of meeting related to the abstractive summary. Note that each sentence in an abstractive summary can be referred to several dialogue acts in the corresponding extractive summary, and each dialogue act can refer to several sentences in the abstractive summary.

4 Extracting Information from Meeting Transcripts

The aim of the paper is to extract information that can be used to derive a meeting summary of a meeting including several participants. We are interested in an approach allowing us to extract the "important" words from a document, which can then be used in further steps, such as summarizing the meeting.

We calculate a weight for each word in a meeting document, where the weight expresses the relative word importance for the meeting with respect to a set of comparison meeting documents. Our hypothesis is that in any meeting words with higher weights are more likely to provide relevant information about and characteristics of the meeting. In this section, we present the two approaches we have evaluated: Term Frequency-Inverse Document Frequency (*tf-idf*), a statistical measure of a word's relative importance, as a reference method, and Non-negative Matrix Factorization, a topic modeling approach that groups words into sets and assigns weights (probabilities) to each topic.

4.1 Term Frequency-Inverse Document Frequency

Term Frequency-Inverse Document Frequency (*tf-idf*) is a measure that gives a higher weight to a term (a word) that appears frequently in a particular document but infrequently in the entire corpus. The term frequency ($\text{tf}(w, d)$) is simply the number of times that a word w appears in a document d. Inverse document frequency (idf) is the logarithmically-scaled proportion of documents in the corpus in which the word w appears. More formally:

$$\text{idf}(w, D) = \log \frac{N}{1 + |d \in D \ s.t. \ w \in d|} \tag{1}$$

where N is the total number of documents in corpus and the denominator is the (laplace-corrected) number of documents d in which a word w appears in corpus D. Thus, the *tf-idf* is:

$$\text{tf-idf}(w, d, D) = \text{tf}(w, d)\text{idf}(w, d)$$

As an example for meeting documents, a stop-word such as "and" may appear often in a document, but not have a high *tf-idf* value because it is repeated in almost every meeting. However, a word such as "lunch" has a high weight, because it appears frequently in a particular document but less often in the entire corpus. *tf-idf* is, for instance, the measure used to quantify terms' importance in [17], and a slight modification is used in [8].

4.2 Topic Modeling: Non-negative Matrix Factorization

There exist various topic modeling methods allowing us to identify the topics present in text documents. Among the considerable number of proposals for probabilistic methods such as Latent Dirichlet Allocation (LDA) or non-probabilistic methods such as Non-negative Matrix Factorization (*NMF*), we focus on the *NMF* in this paper [11]. Indeed, preliminary experiments (not given in this paper) showed that algebraic approaches perform better than probabilistic methods on our meeting documents.

In an *NMF* context, we assume that the matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ represents m unique words in a corpus of n meetings (documents). The goal is to decompose the $\mathbf{A}$ matrix into two non-negative matrices such that $\mathbf{A} \sim \mathbf{WH}$. The columns in $\mathbf{W} \in \mathbb{R}^{m \times k}$ are the topics and the rows are the words represented in the topics. Each item in matrix identifies a non-negative weight for a word in relation to a topic. The matrix $\mathbf{H} \in \mathbb{R}^{k \times n}$ indicates to what degree a meeting is related to the k of topics.

5 Preliminary Experiments with AMI Corpus

To evaluate the effectiveness of the two methods in treating meeting data, we perform several different experiments. In a first experiment, we assess whether *tf-idf* can be used to identify the characteristics of particular meetings when compared to other meetings in the same context. In a second experiment, we evaluate whether *tf-idf* and *NMF* can be used to help in *grouping* meetings. Finally, we assess the agreement of extracted sets of words with the two types of existing summaries (extractive and abstractive) for the meetings in the AMI corpus.

5.1 AMI Corpus Preparation

We evaluate the effectiveness of our proposed techniques on the AMI data set [1,2] containing multi-party dialogues[3]. The AMI data set delivers its meeting documents in different divisions such as divided meeting in terms of words, time, speakers and etc. We generated the plain text files manually using these segmentations and divisions which are available online[4].

[3] http://groups.inf.ed.ac.uk/ami/download/.
[4] https://github.com/pegahani/AMI-prep.

To study the scenario-based meetings, we first classify meeting files w.r.t their subject. According to [2], each 3 to 5 meetings documents are acted out by a single group speaking about the same scenario, divided into several parts. For this reason, we regroup the 138 scenario-based meetings into 34 "effective" meeting blocks. On the other hand, there exist 33 non-scenario based meetings as well that will be studied in our future works.

5.2 Characterizing Individual Meetings

As we have described before, the underlying motivation for this work is to help summarize meetings, in particular identifying decisions taken, action items defined, problems identified, and responsibilities assigned. The scenario-based AMI meetings are intended to simulate the trajectory of a particular project (designing a remote control) from start to finish, condensed into a single day. All 34 groups acting out this scenario follow the same script but will, of course, vary in the concrete implementation of the script. However, if there is something inherent in each stage of the project, i.e. each individual meeting of the same block, we would expect those characteristics to show up in the terms extracted by *tf-idf*.

To evaluate our hypothesis, we use the following experimental setup: we treat a given block of meetings, i.e. meetings acted out by the same group, as the corpus and identify for each individual meeting in this block the twenty highest-scoring words according to *tf-idf*. For each of these words, we then count how often it occurred in the set of words derived from the same stage, e.g. we gather the sets of words derived from the first meeting of each block (M_1) and count duplicates. Table 1 shows the results for each stage, after thresholding word frequency at 4, i.e. 10%, rounded up[5]. We remind the reader that we did not use stemming for those texts. The number at the beginning of each cell therefore reports the modulo stemming, e.g. "cats" counts towards "cat", the number in parentheses the actual count of the word. There are 34 different groups that acted out the scenarios, which means that, for instance, the term "animal" was used in 73.5% of all first meetings of this scenario.

Table 1 (including 10% of twenty highest scoring words for the whole 34 meeting blocks) demonstrates most of the words are appeared in the first stage or the last one while the two stages in the middle have a small share. For instance in M_1 stage, "animal", "cat" and "favourite" have repeated 29, 19 and 12 times respectively. But in the M_2 stage less words appear, for instance "age", "lunch" and "teletext" have appeared each for 9 times. We see that the first and last meetings of a block have more related words than the second and third meetings. This shows that the script aligns the different groups quite closely in the beginning and in the end, but that they diverge in the middle two meetings. The first meeting seems always to be a brain-storm related to animals, and the fourth meeting about how to evaluate the success of the project. In the case of the third

[5] For full results, we refer the readers to: https://github.com/pegahani/Event_detection/blob/master/result/result_4_4.txt.

Table 1. The most often repeated high-scoring words (according to *tf-idf*) for individual meetings in scenario-based blocks

M_1	M_2	M_3	M_4
25: animal	9: age, lunch, teletext	8: solar, wood	17: criteria
19: cat (15)	8: percent, young (4)	6: titanium	13: seven
12: favourite	6: pay	5: spongy, concepts	12: evaluation
11: tool (5), dog (7)	5: settings, zap, seventy, users	4: dark, sample, doublecurved, materials, sensor, banana, circuit, cases, vegetables, fruit	9: sample
7: training, draw	4: set, messages, group, mode, infrared, recognition, speech		8: special, false
6: rabbit, profit, fish			6: evaluate, process
4: friendly, bird, tail, whiteboard, width, characteristics, elephant, morning			5: prototype, leadership, creativity, under
			4: scale, single, fifteen, team, average, budget, curve

meeting, we can speculate that the discussion turned around what materials to use, and the presence of "lunch" as a highly-ranked word in the second meeting reminds us that this meeting took place at the end of the morning.

5.3 Grouping Meetings

Earlier, we have contrasted individual meetings against other meetings within a particular process. In the real world it's not clear how to group meetings, however. In the best case scenario, whoever starts the meeting recording would indicate what project or issue the meeting is related to but we cannot rely on this information being available. A slightly weaker assumption relies on participants stating their names. For example, if *Ada, Billy, Christine* and *Dolores* work together on one project, and *Elise, Frank, Gerd* and *Heather* on a second one, we would assume that certain combinations of names identify *groups* of meetings belonging together.

We therefore change the underlying document for calculating the *tf-idf* score, merging all meetings of a single block into one document, and treating the other merged blocks as comparison documents. The second column from the left of Table 2 shows examples of what happens when we contrast the 3–5 meetings enacted by a particular group against *all other meetings* in the corpus and as we expected, several of the extracted terms are names[6]. When contrasting individual sessions in a block against the other ones in the same block, as in the preceding section, names weren't highly ranked, which is to be expected given that the entire scenario is acted out by the same group of participants.

This indicates that one can contrast meetings against the entire available corpus and group them with other meetings showing the same combination of

[6] For full results, we refer the readers to: https://github.com/pegahani/Event_detection/blob/master/result/result_4_block_scen.txt.

Table 2. Example of words extracted using *tf-idf* t (left) and *NMF* (right) for the first five scenario-base meeting blocks

meeting block	Extracted words using *tf-idf*	Extracted words using Topic Modeling (*NMF*)
meeting block 1	**matthew**, **mael**, **anna**, exce, ip, doctor, decline, assemble, streamed, customizing, asian, voter, undes, **nanne**, highperformance, fik, protec, underlie, provin, zebras	keys, **matthew**, browse, innovation, functionalities, v_c_r_, **mael**, **anna**, sixteen, perfect, demographic, r_c_, receiver, surf, present blinking, cents, store, movie, presented
meeting block 2	incremental, linear, orangutan, barks, squarey, lightening, caramel, computation, parameter, shocks, selled, multidevice, dommage, lawnmower, chec, discus, orangutans, fulf, buck, **olivier**	access, management, incremental, whistle, participant, receive, pear, ami, advantage, ergonomic, define, commands, fulfil, robust, technological, command, cent, task, continue, financial
meeting block 3	**pedro**, midmarket, backlit, crossover, exclusivity, trainable, silvers, **pedros**, uniqueness, paperwork, scheduling, offhand, ditching, consciousness, axes, marketability, ballpark, twelvefifty, synergy, advantageous	cradle, create, niche, sorta, **pedro**, locator, unique, terms, identified, flip, televisions, nah, environmentally, management, mode, shell, marketable, lapel, ta, perspective
meeting block 4	mushroom, **jordan**, coarse, baba, alimentation, mush, gestures, **kemy**, institute, **frahan**, **florent**, laser, longmund, sleeping, ada, saucer, trois, ecological, hmmm, eatable	controller, mushroom, gesture, google, pineapple, powerful, **david**, base, wireless, traditional, lemon, **jordan**, wooden, sophisticated, wire, vocal, participant, ball, bulb, recognise
meeting block 5	turbo, **mando**, panther, indian, spiders, spider, trunk, capsicum, outlier, pepper, kitsch, granularity, playback, spotting, templates, ons, hunt, stylised, epinions, permanantly	station, handy, turbo, **mando**, basis, base, technologies, targeting, wheels, coffee, elephant, sitting, elephants, phrase, milan, instance, morning, crazy, traditional, recognize

names, if they have not been grouped together by the person in charge of recording. The results need improvement, however: while we know that each scenario was acted out by four people, the average number of names recovered by *tf-idf* is only 2.44, with a standard deviation of 1.44. Main contributors to this standard deviation are blocks 25, 27, and 31, neither of which gave rise to a single name. It is possible that participants in those blocks did not state their names (often), a situation that might arise in real-world situations, especially if a certain group of people has already been working together for a while.

In this context, we would also like to point out an interesting observation in our experiments: for the block grouping $M_{52}, M_{53}, M_{54}, M_{55}$, one of the highest-ranking words is "shoulda". This is a non-formal contraction of "should have" and the fact that it is tagged as having one of the 20th highest scores for that block (actually the 8th-highest) indicates that certain participants' *manner of speech* might be enough to group meetings. Other contractions that we have extracted include "you'll" and "ain't". We intend to explore this phenomenon in the future.

As a comparison approach to using *tf-idf* on merged meetings, we also evaluated *NMF*. We use *NMF* to assign a topic to each scenario-based meeting block. As has been mentioned earlier in this section, there are 34 scenario based meeting blocks in total. For this reason we use *NMF* topic modeling with exactly 34 topics to classify the scenario-based meeting blocks, i.e. each block of meetings is assigned to exactly one topic.

To implement the *NMF* method, we used the gensim package [15]. After classification, each meeting should belong to a topic class. As for *tf-idf*, each topic is represented by the 20 most important candidate words according to the induced model. Referring to Table 2, the extracted words for five meeting blocks 1 to 5 is given in the last column[7]. While we can recover names using *tf-idf*, this is not reliably the case when using *NMF* – the average number of recovered names is 0.88, with a standard deviation of 0.91. This means that the information derived the *tf-idf* score and the topic model are complementary. Interestingly enough, when looking at meeting block 4 in Table 2, we find "david" and "jordan" in the *NMF* results, who refer to the *same* person in the acting group.

5.4 Quality of Extracted Sets of Words

In order to evaluate the extracted information of the meetings, we compare them with the abstractive and extractive summaries as follows.

Given a set of words S extracted from a meeting M, we define the accuracy S w.r.t the abstractive summary (extractive summary) of M, $s_{abs}(M)$ ($s_{ext}(M)$), as:

$$acc_{abs}(S, M) = \frac{|S \cap s_{abs}(M)|}{|S|} \; (acc_{ext}(S, M) = \frac{|S \cap s_{ext}(M)|}{|S|}), \text{ i.e.,}$$

number of words of S that appear in the abstractive (extractive) summary divided by the number of words in S. We match words modulo stemming, for instance, if S contains "painting" and s_{abs}/s_{ext} a similar word such as "paint", the word is considered to have appeared.

We assess the quality of the extracted sets of words in two ways: Figs. 1 and 2 show how sets of words extracted as described in Sect. 5.2 perform w.r.t. to abstractive and extractive summaries.

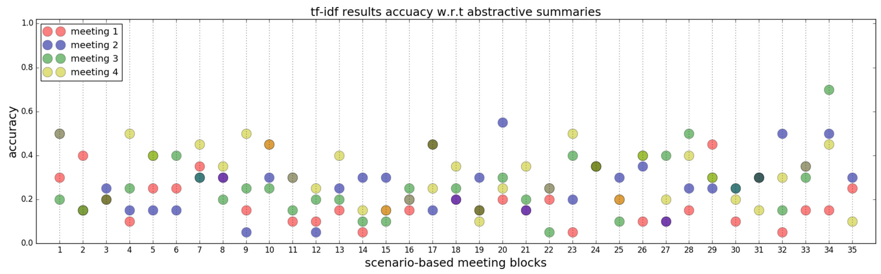


Fig. 1. Abstractive accuracy for sets of words extracted from individual meetings, compared to the other meetings in the same block, using *tf-idf*.

In overall, abstractive accuracy results are always less than 0.6 but in many cases they vary between 1.0 to 4.0. It is noticeable that for almost half of the blocks, the abstractive accuracy for the first meeting is lowest. As we saw in

[7] For more results you can visit: https://github.com/pegahani/Event_detection/blob/master/result/Topic_modeling_nmf_block_34_topics.txt.

Table 1, the brain-storming in the first meeting involved concrete animal names that are not necessarily presented in the abstractive meeting.

Most extractive accuracies are above 0.6 and noticeable cases have the accuracy more than 0.8. As expected, extractive accuracy is higher than the abstractive one. This is because our accuracy measure computes the percentage of words appearing in each summary. Since the extractive summary includes more words as a subset of the meeting documents, the accuracy value for extractive summaries is far higher than the abstractive one. In addition, abstractive summaries by definition *abstract* from the actual contents of the meeting.

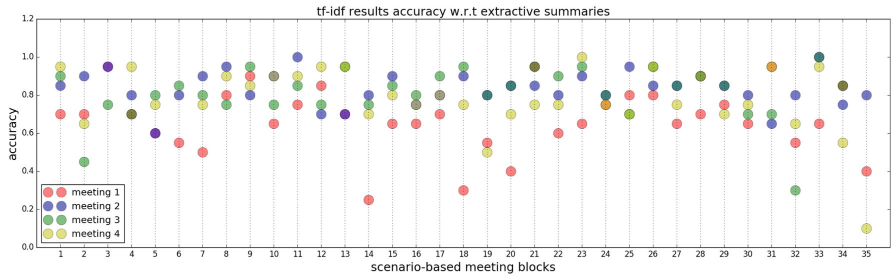


Fig. 2. Extractive accuracy for sets of words extracted from individual meetings, compared to the other meetings in the same block, using *tf-idf*.

The second assessment uses the sets of words extracted according to the description in Sect. 5.3, and accuracies are calculated w.r.t. the *merged* summaries of all meetings in a block.

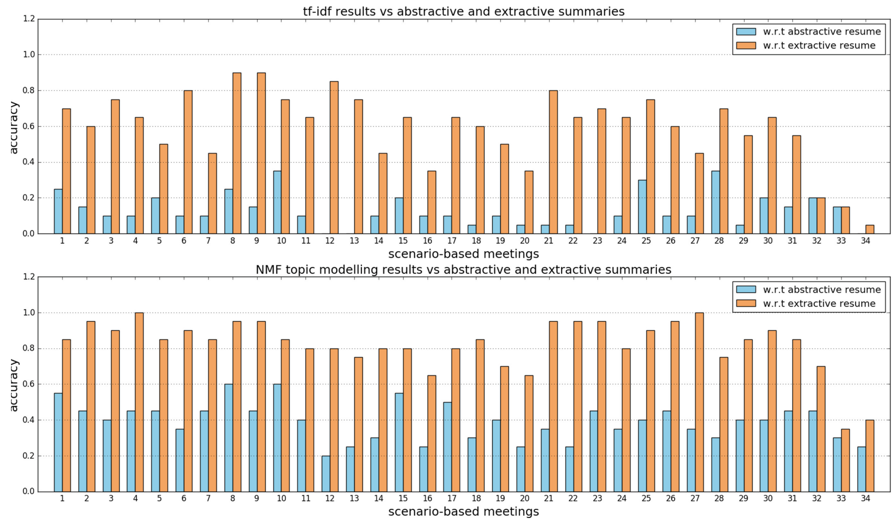


Fig. 3. The upper graph (lower graph) indicates the abstractive and extractive accuracies for sets of words extracted from scenario-based meeting blocks using *tf-idf* (*NMF*).

The upper graph in Fig. 3 reports the accuracies of sets of words extracted from each meeting block using *tf-idf*. Similar to the preceding results, extractive accuracy is higher than the abstractive one. According to the graph, the extractive accuracy is more than 0.6 for the extractive summaries while the abstractive accuracy is less than 0.4 for all the scenario-based meeting blocks.

The lower graph in the figure shows abstractive and extractive accuracies for sets of words extracted by *NMF*. By comparing this graph with the *tf-idf* results, we see that *NMF* gives better results for both summary types. This is the flip side of *NMF*'s inability to pick out names – instead of picking up particularities of the group – *NMF* learns topics describing the meeting block, matching summaries better.

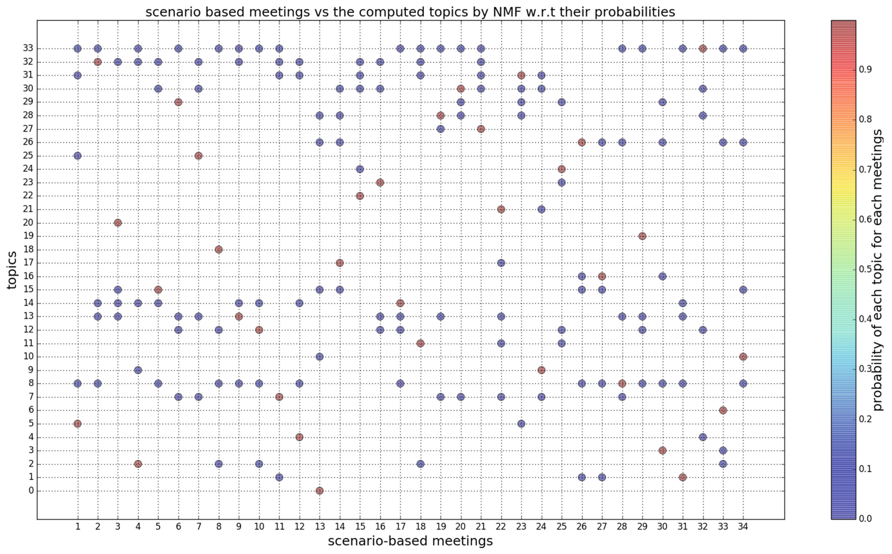


Fig. 4. The x axis indexes the 34 scenario-based meeting blocks while the Y axis indicates the list of 34 topics. The figure shows the 5 highest-ranked topics for each meeting block. The heat map on the right hand side maps probabilities to the colors used in the plot.

Since topic modeling methods compute the probability with which each text is related to each topic, Fig. 4 shows for each meeting block how probable it is that it concerns a particular topics. The color bar in the right side of the figure shows the probabilities i.e. a point with a color closer to red is more highly probable. For instance, meeting 1 concerns topic 5 with a probability of more than 0.9 while, it is about topics 8, 25, 31 and 3 with a very low probability (less than 0.1). It shows that there are no ambiguities – each meeting block has one topic assigned to it with very high probability (in excess of 0.9), and all others are at less than 0.1 probability.

6 Conclusions and Perspectives

In this article, we tested representatives of two paradigms for term extraction from documents – numerical statistics and topic modeling – to extract important information from texts in the context of recorded and transcribed multi-party dialogues, more specifically the AMI data set. We evaluated three settings: characterizing individual meetings in comparison to other thematically related meetings, identifying names of participants tying together related meetings, and matching extracted sets of words to provided gold-standard summaries.

In terms of the first setting, *tf-idf* performs rather well, consistently characterizing first and last meetings of a scenario, and showing good extractive accuracies (and to a lesser degree abstractive ones), i.e. comparisons of the extracted sets to gold-standard summaries. When it comes to identifying names, *tf-idf* shows better performance than *NMF* but they are arguably not good enough to reliably group meetings, explicit tagging before recording is therefore probably required.

Topic modeling via *NMF*, on the other hand shows better performance w.r.t the provided gold standard (abstractive and extractive summaries) when we attempt to characterize a full scenario, i.e. a block of thematically related meetings acted out by one group.

A conclusion to be drawn from our results is therefore that the two approaches complement each other, providing results that are useful for different aspects of the larger task of characterizing and summarizing meetings. To continue in this direction, automatically grouping meetings seems to be the most important issue to improve on because working on related meetings is foundational for other the other tasks, i.e. characterizing different types of meetings and/or identifying trends that help in understanding the progression of a particular project. A second question is how to arrive at summaries from terms sets – the process of extracting sentences related to extracted terms, and translating those into abstractive summaries in turn, is not an obvious one but needs to be tackled.

Acknowledgement. This work is supported by the FUI 22 (REUs project) and the ANR (French Research National Agency) funded project NARECA ANR-13-CORD-0015.

References

1. Augmented multi-party interaction (2010). http://www.amiproject.org
2. Carletta, J.: Unleashing the killer corpus: experiences in creating the multi-everything AMI meeting corpus. Lang. Resour. Eval. **41**, 181–190 (2007)
3. Fernández, R., Frampton, M., Dowding, J., Adukuzhiyil, A., Ehlen, P., Peters, S.: Identifying relevant phrases to summarize decisions in spoken meetings. In: Proceedings of Interspeech 2008, Brisbane (2008)
4. Fernández, R., Frampton, M., Ehlen, P., Purver, M., Peters, S.: Modelling and detecting decisions in multi-party dialogue. In: Proceedings of the 9th SIGdial Workshop on Discourse and Dialogue, SIGdial 2008, pp. 156–163. Association for Computational Linguistics, Stroudsburg, PA, USA (2008)

5. Galley, M., McKeown, K., Fosler-Lussier, E., Jing, H.: Discourse segmentation of multi-party conversation. In: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics, ACL 2003, vol. 1. Association for Computational Linguistics, Stroudsburg, PA, USA (2003)
6. Georgescul, M., Clark, A., Armstrong, S.: Exploiting structural meeting-specific features for topic segmentation. In: TALN/RECITAL, Toulouse (France), pp. 15–24 (2007)
7. Gurin, Y., Szymanski, T., Keane, M.T.: Discovering news events that move markets. In: Intelligent Systems Conference 2017 (IntelliSys2017), London, United Kingdom, 7–8 Sept 2017 (2017)
8. He, Q., Chang, K., Lim, E.P.: Analyzing feature trajectories for event detection. In: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2007, pp. 207–214. ACM, New York (2007)
9. Kleinberg, J.M.: Bursty and hierarchical structure in streams. Data Min. Knowl. Discov. **7**(4), 373–397 (2003). https://doi.org/10.1023/A:1024940629314
10. Lau, J.H., Collier, N., Baldwin, T.: On-line trend analysis with topic models:#twitter trends detection topic model online. Proc. COLING **2012**, 1519–1534 (2012)
11. Lee, D.D., Seung, H.S.: Learning the parts of objects by nonnegative matrix factorization. Nature **401**, 788–791 (1999)
12. Purver, M., Dowding, J., Niekrasz, J., Ehlen, P., Noorbaloochi, S., Peters, S.: Detecting and summarizing action items in multi-party dialogue. In: In Proceedings of the 9th SIGdial Workshop on Discourse and Dialogue (2007)
13. Purver, M., Griffiths, T.L., Körding, K.P., Tenenbaum, J.B.: Unsupervised topic modelling for multi-party spoken discourse. In: Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics, ACL-44, pp. 17–24. Association for Computational Linguistics, Stroudsburg, PA, USA (2006)
14. Ramage, D., Dumais, S.T., Liebling, D.J.: Characterizing microblogs with topic models. In: ICWSM, vol. 10(1), pp. 16 (2010)
15. 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. ELRA, Valletta, Malta (2010). http://is.muni.cz/publication/884893/en
16. Riedhammer, K., Favre, B., Hakkani-Tür, D.: Packing the Meeting Summarization Knapsack. In: Interspeech, Brisbane (Australia). Unknown, Unknown or Invalid Region (2008). https://hal-amu.archives-ouvertes.fr/hal-01194290
17. Sayyadi, H., Hurst, M., Maykov, A.: Event detection and tracking in social streams. In: In Proceedings of the International Conference on Weblogs and Social Media (ICWSM 2009). AAAI (2009)
18. Tur, G., et al.: The CALO meeting speech recognition and understanding system. In: 2008 IEEE Spoken Language Technology Workshop, pp. 69–72 (2008)
19. Tur, G., et al.: The calo meeting assistant system. IEEE Trans. Audio Speech Lang. Process. **18**, 1601–1611 (2010)
20. Weng, J., Lee, B.S.: Event detection in twitter. In: ICWSM, vol. 11, pp. 401–408 (2011)

Automatic Matching and Expansion of Abbreviated Phrases Without Context

Chloé Artaud[1,2], Antoine Doucet[1(✉)], Vincent Poulain D'Andecy[3], and Jean-Marc Ogier[1]

[1] L3i, University of La Rochelle, La Rochelle, France
{chloe.artaud,antoine.doucet,jean-marc.ogier}@univ-lr.fr
[2] Direction Générale de l'Armement, La Rochelle, France
[3] Yooz, Aimargues, France
Vincent.PoulainDAndecy@yooz.fr

Abstract. In many documents, like receipts or invoices, textual information is constrained by the space and organization of the document. The document information has no natural language context, and expressions are often abbreviated to respect the graphical layout, both at word level and phrase level. In order to analyze the semantic content of these types of document, we need to understand each phrase, and particularly each name of sold products. In this paper, we propose an approach to find the right expansion of abbreviations and acronyms, without context. First, we extract information about sold products from our receipts corpus and we analyze the different linguistic processes of abbreviation. Then, we retrieve a list of expanded names of products sold by the company that emitted receipts, and we propose an algorithm to pair extracted names of products with the corresponding expansions. We provide the research community with a unique document collection for abbreviation expansion.

Keywords: Abbreviations · String distance · Information extraction

1 Introduction

In the general objective of detecting false documents by verifying the information contained in the document, we must extract and model this information in order to be able to compare it with information that we will search for in external resources. However, administrative documents, such as payslips, administrative forms, invoices, proofs of purchase, have many constraints, particularly with regard to the document layout and the space available for each bit of information. Indeed, administrative documents, except for letters and contracts, are intended to be sufficiently concise and clear to be analyzed at first glance by humans.

This conciseness is often expressed by a layout containing precise tables and locations for the different elements specific to each type of document. For instance, in an invoice, we will often find a logo in the upper part, a header containing metadata, a table with names of products or services on the left, prices on the right, a total amount below, and the fine print at the end of the document. This highly organized structure imposes spatial constraints on textual and graphical elements: all the information necessary for the usefulness of the document must fit into a single document, often a single page for practical reasons. While logos and other graphic elements can be reduced, the textual information must be shortened while remaining sufficient to be understandable.

This constraint, common to many document types, allows us to observe the linguistic phenomenon of reduction. These abbreviations, that appear in many forms, are a bottleneck for the automatic analysis of document information. Indeed, it is very difficult to extract and interpret information that is not exhaustive and not fully explicit. We must therefore seek to associate abbreviations with their full form, or "expansion".

The task of associating abbreviations with their possible expansions is not simple because it involves many constraints: the absence of natural language context, the diversity of the types of abbreviations, the double level of abbreviation in words and in phrase, the ambiguity of the product names.

In Sect. 2 of this paper on related work, we show that the context-less automatic matching of abbreviations and their complete forms is a problem that has not been much studied in the state of the art. Section 3 presents our data set and analyses the different type of shortened forms observed. Section 4 explains the proposed approach for pairing abbreviated phrases and their possible expansions. Section 5 describes and discusses the results of this approach. We finally conclude and provide perspectives in Sect. 6.

2 Related Work

2.1 French Linguistics Definitions

French linguistics grammar does not much describe abbreviations in the French language, their socio-linguistic and semantic uses and values. For example, Riegel et al. only grant two pages and is focused in truncation and initialism only [1]. The use of abbreviations are anecdotal to this work as it only belongs to the scriptural aspect of the language and is specific to each writer. To define the abbreviations, it is the graphic aspect that also prevails for Grevisse and Lits [2], except for the phenomenon of reduction, which is the construction of a new form from another having the same meaning, by removal of a part of the full form. Similarly, acronyms enter the current language by becoming a word, as for instance "AIDS" in English. Grevisse and Lits [2] further classify measurement units as symbols, not abbreviations, noting that units (in metrology or chemistry, for example) have lost their abbreviation value to take a symbolic value and are no longer followed by periods.

While the abbreviations in these two reference works are purely personal graphic practice, for Martinet [3], on the contrary, the abbreviations are used to communicate while providing the least effort. In the case of abbreviations, it is a question of keeping the strict necessary to transmit information and thus saving memory, time and space. The abbreviation is an integral part of the evolution of the language: the more frequently an expression is pronounced, the more likely it is to be abbreviated, by eliminating non-specific elements, by truncation or by initialism.

2.2 Abbreviations Disambiguations

To our knowledge, little work exists in Natural Language Processing on the matching of all types of abbreviations with their expansions. However, we can find many works on the extraction of acronyms (often confused with the term "abbreviations") in texts and their disambiguation according to context. This application is particularly present in Biology and Medicine, where acronyms are very numerous, both in academic articles, especially on MEDLINE[1], and in clinical reports.

These studies [4–6] first attempt to extract abbreviations from the text. Authors use pattern-matching rules for mapping an abbreviation to its full form: acronyms are often capitalized, and sometimes in brackets when they appear for the first time [7]. They then use context-based machine learning methods to choose the right expansion from a number of different dictionaries of acronym-expansions pairs, as explained by Gaudan et al. [8]. These methods achieve very good results, but are not applicable to our data.

2.3 String Distance Methods

Our data are only strings of abbreviated forms and expansion forms, without context. For this reason, we can only work with string-based methods. The best known metric for calculating a distance between two strings is the Levenshtein distance (also known as the edit-distance) [9]. It gives the smallest number of character changes (deletion, insertion, substitution) to switch from one string to another. The weaker the result, the less differences between strings. Numerous variants have been proposed for this metric, in order to take into account also the transpositions (Damerau-Levenshtein) or only substitutions in strings of the same length (Hamming).

In our problem, the strings are of different lengths and only the inserts are of interest to us, at level of characters in words and at level of words in phrases. For this second step, we can place the problem in set theory, considering the phrase as a set of words, which allows us to use measures such as the Jaccard index or the overlap coefficient. The Jaccard index is a simple similarity coefficient between two sets, defines as the size of the intersection divided by the size of the union. This index is between 0 (nothing in common) and 1 (identical sets).

3 Data Collection

We manually created a corpus of 248 product names as typed on sales receipts from a single store. These product names, constrained to be comprehensible for

[1] Medical Literature Analysis and Retrieval System Online, https://www.nlm.nih.gov/pubs/factsheets/medline.html.

human while being limited to 20 characters, contains many reductions, both at level of words and at level of phrase, and many other obstacles to matching short and long forms.

3.1 Reductions

Abbreviations. In our receipts corpus, the abbreviation is a graphic phenomenon: the word is written in a short form by deleting some of its letters. We find two types of abbreviation in our corpus:

- Apocope: removal of the end of the word
  Apocope abbreviations may or may not be followed by a period (1a) and (1b). We notice that the period is followed by a space only in one case in our corpus. The rest of the time, it serves to separate a short form of a word from another word, in its short or long form. Our corpus contains a particular apocope type, that is not only a graphic phenomenon, but also phonetic and linguistic phenomenon: the word in its shortened form is so much used that it now exists in everyday language. This is the case of "BIO", which is an abbreviation of "*biologique*" ("from organic farming").
- Syncope: deletion of letters inside the word
  Syncope abbreviations also may or may not be followed by a period. We note that sometimes the first and last letters are kept (2a), but more often it is three representative consonants (2b). In other cases, some vowels are deleted in the word (2c).

These different forms of abbreviations present a particular challenge for their recognition: it seems that there are no systematic rules, or pattern, for the construction of abbreviations in this corpus, apart from the presence of the first letter of the word. Table 1 shows some examples of abbreviations, with the expansion of every word, the phrase expansion, the expansion of the corresponding expression in our corpus of expansions and a translation into English.

Initialisms. The first letters of each word of a compound word or of a brand name composed of several words are assembled. If the initialism is pronounced as a word, we consider it as an acronym. In our corpus, initialisms are sometimes followed by a period, contrary to the typographical rules.

Table 2 shows a few examples of initialisms of our data. We can notice that in the three examples, initialisms take the first letter of the prepositions "de" ("*of*") and "à" ("*to*") contrary to the use.

Symbols. Words are sometimes represented by (or contain) mathematical symbols. We thus frequently find the symbols "+" and "/". These special characters create segmentation problems. Their heterogeneous use does not allow systematic treatment. Indeed, in Table 3, we can see that the slash character can separate words (abbreviated or not) to mean a mixture of ingredients (1), whereas

Table 1. Examples of abbreviations

Type	Abbreviation	Words expansion, Expansion and *English gloss*
(1a)	**CONF.FIG.PROV.R.**FR	*CONFiture FIGues PROVence Reflet FRance*
		Confiture de figues de Provence Reflets de France
		Fig jam from Provence Reflets de France
(1b)	**TAB CHO** CRF NR **BIO**	*TABlette CHOcolat CaRreFour NoiR BIOlogique*
		Chocolat bio noir 74% cacao Carrefour Bio
		Organic chocolate 74% cocoa Carrefour Bio
(2a)	**PT** BEUR.CHO **LT** NOI	*PetiT BEURre CHOcolat LaiT NOIsette*
		Biscuits petit beurre chocolat lait Petit Ecolier
		Butter cookies with milk chocolate Petit Ecolier
(2b)	160G **BLC PLT** 4TR.F	*160 Grammes BLanC PouLeT 4 TRanches Fines*
		Blanc de poulet Carrefour
		Carrefour Chicken Breast
(2c)	**PT** DEJ.FRUITS **RGES**	*PetiT DEJeuner FRUITS RouGES*
		Biscuits petit déjeuner aux fruits rouges Carrefour
		Breakfast biscuits with red berries Carrefour

Table 2. Examples of Initialisms

Initialisms	Expansion and *English gloss*
2X30CL VELOUT.**PDT.**	*Velouté de Pomme De Terre*
	Cream of potato soup
PT L'EVEQUE **RDF**	*Pont l'Eveque Reflet De France*
	Pont l'Eveque Reflet de France
YAF PANIER FRAM/MU	*Yaourt Aux Fruits Panier de Yoplait framboise/mûre*
	Fruit yoghurt Panier de Yoplait raspberry/blackberry

in (2), the slash intervenes between two words (preposition and noun) of the same unit of meaning. It should be noted that this abbreviation "s/s" may correspond to the bigram "without sugars" but also "without salt" in our corpus. The slash is also observed in its signifying form of mathematical symbol (3). In this last case, we can not define the slash as a separator because the digit-slash-digit set, a mathematical fraction, constitutes a single semantic unit: "half".

Other mathematical symbols are frequently used in cash register abbreviations, such as the plus sign and the letter X, used in place of the multiplication cross. In (4), the sign "+" has a true mathematical value: it is to signify the addition of a numerical quantity to the quantity indicated before the character. On the other hand, in (5), the "+" character is inserted between two (abbreviated) words, denoting a product consisting of two entities. The "+" character represents, in both cases, the combination of two units that are not usually assembled, however, the treatment to be performed is not the same in the two examples. In the first case the character must be interpreted as a word to disambiguate, while in the second, the character must be considered as a separator not to be included in the words framing.

Table 3. Examples of Symbols

Type	Initialisms	Expansion and *English gloss*
(1)	BURGER EMM/CHEDDAR	*Burger emmental* ***et*** *cheddar*
		Burger emmental **and** cheddar
(2)	KRISPROLL.S/S	*Krisprolls* ***sans sucre***
		Krisprolls **without sugar**
(3)	BEURRE BIO 1/2 SEL	*Beurre bio* ***demi****-sel*
		Half-salt organic butter
(4)	CHOCOLATINE X4 + 1 G	*4 chocolatines* ***et*** *une gratuite*
		4 chocolate croissant **and** one free
(5)	DUO TERRIN+MOUS.CD	*Duo terrine* ***et*** *mousse de canard*
		Duo terrine **and** duck mousse

Numbers. The mixture of digits and letters is particularly problematic in our data because we can not apply a general rule to treat it. Indeed, there are several cases where numbers and letters are mixed:

– Expressions of quantities, associating a number and the abbreviation of a standard measurement unit (85G : "85 g")
– Expressions of quantities, associating a number with an abbreviation of a word or a complete word (4FRT : "4 fruits", 4FROMAGES : "4 cheeses")
– Number of units, represented by the letter "X" (here meaning "times") contiguous before or after the number (4X125G : "4 units of 125 g")
– Abbreviations mathematically symbolized, associating a fraction and letters (1/2ECR : "semi-skimmed")
– Names of products or brands containing numbers and letters, which should not be separated (T45 : "type of flour")
– Absence of spaces between letters and numbers (75 VDP OC RG BIO09 : "organic red wine of year 2009")

3.2 Other Problematic Features

While the different forms of reduction in the corpus present issues of segmentation and analysis of the text, other characteristics, specific to the format of the document, must be highlighted. Indeed, the word segmentation of the receipts is made difficult by various typographic elements, such as the case used, the presence of periods of abbreviations, the absence of spaces or the concatenation of numbers and units of measure. In rare cases, no character (space, point) can identify the break between two words. In the following example, characters "X" and "G" following a number are lexical units themselves and have to be separated from the abbreviation "DESS.".

Example 1. Product name : 4X100GDESS.PANACHE
Segmentation : 4 X 100 G DESS. PANACHE
English gloss : *four times one hundred grams of mixed dessert cream*

Typographic Elements

Letter case. The characters of this corpus are all capitalized, which does not allow to easily visualize initialisms, as it is the case in most works on acronyms and abbreviations in natural language texts. Nor does it make it possible to distinguish proper names, such as brand names for example, which could have helped in the analysis of documents.

Diacritics. The corpus does not present any diacritic sign, that is, no accent or cedilla. This absence is also a handicap for the analysis because it can cause the ambiguity of certain words, such as PATE, which can correspond to *pâte* ("paste") or *pâté* ("pâté").

Punctuation. Some product names have dots that act as both segmentation characters and truncation signs. In the example 8XDOS.SENSEO CAPP., the words DOS (for *dosette*, "capsule") and SENSEO (brand) should be separated while integrating the period with the word preceding it, contrary to the space, to have the possibility to force the search for the long form of the word thereafter. Indeed, in this example, the lowest distance finds the word *dos* ("back") instead of the word *dosette*, which is impossible because of the presence of the point which means that the word has been abbreviated. However, segmentation should not be applied to the period between two digits, as in the example CRISTALINE 1.5L because it is a decimal number.

Pseudo-syntactic Elements. Beyond the word scale reductions and problematic typographic elements previously explained, we also need to analyze reductions at phrase level, which we can consider as multi-word expressions, to allow us to better understand the difficulties of alignment between short forms and long forms. These elements concern the reduction processes used, not from a lexical point of view but from a syntactic and semantic point of view.

Ellipsis. Product names usually include only names and adjectives necessary for product distinction. Prepositions and determinants are omitted, except in rare cases. These ellipses do not prevent the understanding of product names, because these words are almost meaningless. However, some ellipses concern words that are more important for automatic understanding because they carry meaning, like in Example 2: it's semantic ellipses.

Example 2. 205G DESSERT NOIR : *Chocolat noir Nestlé Dessert* ("Dark Chocolate Nestlé Dessert")

Random Word Order. The order of words in product names is not always intuitive (300G MOUSSAKA BARQ instead of *Barq[uette de] Moussaka*, "tray of Moussaka"). Similarly, the order of words in similar products is not always the same.

Ambiguity of Reductions. The reductions can be ambiguous, as we have seen in the state of the art. Thus, we find in our corpus some identical reductions that do not correspond to the same long form. For example, we find PDT for initials of *Pomme de terre* ("Potato") and for syncope of *Président* ("President").

Several Reductions for a Single Long Form. For some recurring words, we have noticed the diversity of reduced forms in receipts. For instance, we can find SLD., SAL, SALAD, SLDE and SALADE for *salade* ("salad"). We can notice that some shortened forms are finally not so much reduced in terms of characters. This is the case for example of PIZZ. which contains as many characters as PIZZA.

Spelling Variants. Receipts also contain graphic or spelling variants of certain words. Sometimes these variants lengthen the word. For some brands, it is easier to simplify the spelling. This is the case of MINIBABY (for "Mini Babybel"), which is both a contraction of two words and an abbreviation.

3.3 Corpus of Product Names (Possible Expansions)

In order to find a less ambiguous form for each reduced forms present on the receipts, we have created a reference corpus containing a list of products whose names are detailed and unabbreviated.

We have thus extracted from the Web a list of 13 888 titles of products distributed by the Drive-in service of a supermarket of the same brand as that of our receipts. The Drive-in service allows you to shop online by choosing products with the information the site provides: detailed product name, price, price per kilogram, product weight, promotions and photography.

The choice of this supermarket brings an advantage and a disadvantage: it allows to have a larger choice of products, and therefore a broader reference corpus, but at the same time, the proposed products are not always the same as those bought in convenience stores. In fact, consumers are not the same in a small local area, located in a student district, where our receipts come from, and in a large area on the outskirts. The products sold are not the same either: we will mainly find in the minimarket meals (salad, sandwiches, ready meals), cupcakes and drinks individually, while we will find more fresh or unprepared products, in family quantity, in the supermarket.

These product names differ a lot from the product names extracted from the receipts. While brevity is required on the receipt, it is detail and accuracy that are most important on the purchase website, to be as exhaustive as possible so that the customer knows which product he buys without having it in front of him. The names of products from the Web are consequently longer (37 characters on average) and contain an average of 5.5 words. Each name product contains a key word, often one or several qualifying terms (adjective or noun) and end with the proper noun of the product or the brand, such as the following example.

Example 3. Céréales goût caramel/chocolat Lion (for "Caramel and chocolate flavored cereals named Lion")

We can however observe some irregularities in these formulas such as repetition of a part of the phrase. There is also some abbreviated words (current initialisms or truncations such as "choco" or "PDT"), ellipses of prepositions ("chocolat lait" instead of "*chocolat au lait*" for "milk chocolate"), numbers and symbols.

4 Proposed Approach

We want to find the best long form for each shortened term of our receipts corpus. Matching a shortened term with its full form without context is a very difficult task, especially when the shortened term is so noisy.

4.1 Automatic Pre-processing

Our previous analysis allows us to note the main differences between our two datasets. First, the shortened form is totally uppercase and does not contain any diacritics. In order to make the source (receipts terms) and the target (web extracted terms) comparable, we normalized all target characters to capitalize them and delete all diacritics.

Then, we have to segment both receipts terms and web terms into words, considering the difficulties explained below concerning symbols and absence of space. According to the case, we consider one or two tokens around a symbol like "+", "/", ".". We also separate in two tokens the numbers followed by more than 2 letters, such as 4FROMAGES ("*4 fromages*" for "4 cheeses"). We choose the 2-letters limit to avoid to separate the numbers followed by symbol of quantity, like in 75CL ("*75 centilitres*" for "75 centiliters").

We also define the list of the most frequent trigrams of the web terms, to create a dictionary of most probable initialisms. Indeed, we previously noticed that all initialisms of our receipts corpus are composed of three letters, and correspond to frequent multi-word expression, such as PDT. Thus, for each 3-letters word in receipt expression, which is not a real word, we try to figure whether it could be an initialism.

4.2 Automatic Matching

Terms of receipts are abbreviated both at the word- and at the phrase-level. This requires to implement a two-step algorithm.

First, we calculate the distance between words of shortened form and each word beginning with the same letter as the product names extracted from the web. We save the words within a short enough distance with the source word, or the initialism. To calculate this distance, we use a weighted Levenshtein distance for the first step, which allows us to put a different weight to each editing operation: substitutions, deletions and insertions. We multiply the cost of each operation by the corresponding chosen weight instead of incrementing by 1. In the case of the search of minimal distance between a shortened form and a long form, we want to have a lower weight for insertion than for the two others. We find after computation that the better weights for deletions and substitutions are 11, leaving insertions with a cost of 1, to obtain the better matching. If no distance is satisfactory, no word is recorded.

For the second step, we propose and compare two different methods. The first one is the weighted Levenshtein distance. From the first step, we obtain a list of probable words that we concatenate to have a phrase we can compare with each web phrase, calculating the distance between them. As for the word level step, we save the expression that have the minimal distance with the source phrase and the best weight is 11. The second one is the Jaccard index, using the intermediary list of words obtained from the first step, as a set. We save the expression that have the higher index (included between 0 and 1).

5 Experiments

5.1 Ground Truth

To evaluate our approach, we create a ground truth, manually associating receipts forms and web forms in a JSON format. We encounter some difficulties, because human does not always understand the reduction form, or cannot disambiguate between two similar products with different brands. For example, the PET 1.25L EAU GAZ form should correspond to one item of the 6 full forms list, each corresponding to a different brand, but we can not know which one, and our approach could not either.

For these cases of impossible disambiguation, we propose all the possibilities in our ground truth, to not have false negatives. In some cases, the products on the original receipts are not sold by the Drive-in service, and thus have no shortened form in our corpus. Such cases are evidently discarded in the evaluation.

5.2 Results and Discussions

Results are measured based on the binary match between the proposed long expression and the ground truth, which can return “True” or “False”, depending on whether the proposed long expression is the correct one or not. In addition, cases when no ground truth is associated to the abbreviation are ignored (cases that were not annotated or cases when even the human evaluator cannot perform the task). Thus we compute a percentage of correct answers, which is 36,87% for the best combination of weights in weighted Levenshtein method, and 33,64% for the Jaccard index method.

We observe that the Jaccard approach does not perform as well as the method based on weighted Levenshtein. This is due to marks of plural on many words: while the edit distance increases by 1, the Jaccard index does not recognize the same word. For example, OIGNON JAUNE, corresponding to *Oignons jaunes* (plural) in web source (for “Yellow onion”), with the intermediary list of words containing [OIGNON (singular), JAUNE (singular)]. The Jaccard index between the intermediary word set and the web word set is zero, resulting in a wrong answer. The weighted Levenshtein gives a score of 2 and provides the correct answer.

The traditional measures, namely Precision, Recall and F-Measure, are not the most appropriate to evaluate the performance of our system, because there is no prediction, only scores corresponding to distances. If we select the criteria $distance = 1$ for correct prediction and $distance \neq 1$ for wrong prediction, we obtain the results presented in Table 4. We only see with the first two results that when the distance is small, the result of the algorithm is always correct, but there are very few such cases. The next two lines show the maximal F-Measure we can find by changing the criteria of selection of relevant elements.

Table 4. Evaluation with traditional measures

Method	Precision	Recall	F-measure	Rejection	Accuracy
Jaccard index = 1	1.0	0.06	0.10	1.0	0.68
Weighted-Levenshtein distance = 1	1.0	0.04	0.07	1.0	0.65
Jaccard index $\geq$ 0.33	0.60	0.79	0.68	0.74	0.76
Weighted-Levenshtein distance $\leq$ 50	0.47	0.69	0.56	0.54	0.59

Given that these evaluation results are not very conclusive, we propose an evaluation approach that allow a more qualitative evaluation of results. Only one third of our shortened phrase corpus is correctly found, but, when we analyze the results, we see that performance is not as weak, since the failed word is often the brand name, and most of the phrase is often correctly identified.

The Table 5 shows the average of the scores of three distances computed between the proposed phrases and the ground truth according to the method used. We can observe that there are only 16 edit operations between the two strings and that almost half of the words of the whole two phrases are common. The third line of Table 5 concerns the following test : we give the receipts forms to a French human and he has to find by himself what it could mean and to write the name of the product. We observe that the distance is more important than with our algorithm. This is explained by the addition of many keywords describing the product, as "*boîte de conserve*" (for "can").

Table 5. Evaluation with distance measures

Method	Success rate	Jaccard	Overlap	Levenshtein
Jaccard	33.64	0.44	0.45	15.98
Weighted-Levenshtein	36.87	0.49	0.43	15.46
Human		0.19	0.06	18.34

6 Conclusion

The heterogeneity of product name reductions, whether in the diversity of reduced forms or more generally in the variety of reduction processes, the presence or absence of punctuation or the mixture of special characters, numbers and letters, makes rich the corpus analysis and complex its automatic processing. The automatic matching of shortened words and phrases with their expansion without context is an undeniable challenge. Indeed, the biggest part of people searching for matching abbreviations and expansions uses context to find relations between them in natural language or to disambiguate the meaning.

Qualitative analysis of our experimental results are very promising and we think that the combination of the presented approach with an automatic analysis of contextual elements (like other products bought, hours of shopping...) would allow us to improve the choice of the better expansion for each shortened form.

To foster further research, we make the dataset and its ground truth available to other researchers within an ICPR 2018 competition[2].

References

1. Riegel, M., Pellat, J.C., Rioul, R.: Grammaire méthodique du français. Presses universitaires de France (2016)
2. Grevisse, M., Lits, M.: Le petit Grevisse: Grammaire française. Grevisse Langue Française, De Boeck Secondaire (2009)
3. Martinet, A.: Éléments de Linguistique Générale ... Collection Armand Colin, no. 340. Section de litérature. Librairie Armand Colin (1967)
4. Yeates, S.: Automatic extraction of acronyms from text, pp. 117–124. University of Waikato, (1999)
5. Yu, M., Li, G., Deng, D., Feng, J.: String similarity search and join: a survey, vol. 10(3), pp. 399–417 (2016)
6. Larkey, L.S., Ogilvie, P., Price, M.A., Tamilio, B.: Acrophile: An automated acronym extractor and server. In: Proceedings of the ACM Fifth International Conference on Digital Libraries, DL 2000, Dallas TX, pp. 205–214. ACM Press (2000)
7. Nadeau, D., Turney, P.D.: A supervised learning approach to acronym identification. In: Kégl, B., Lapalme, G. (eds.) AI 2005. LNCS (LNAI), vol. 3501, pp. 319–329. Springer, Heidelberg (2005). https://doi.org/10.1007/11424918_34
8. Gaudan, S., Kirsch, H., Rebholz-Schuhmann, D.: Resolving abbreviations to their senses in medline. Bioinformatics **21**(18), 3658–3664 (2005)
9. Levenshtein, V.I.: Binary codes capable of correcting deletions, insertions, and reversals. Soviet physics doklady. **10**, 707–710 (1966)

[2] Fraud Detection Contest: Find it!: http://findit.univ-lr.fr.

Lexical Resources

Relation Extraction Datasets in the Digital Humanities Domain and Their Evaluation with Word Embeddings

Gerhard Wohlgenannt[1](✉), Ekaterina Chernyak[2], Dmitry Ilvovsky[2], Ariadna Barinova[1], and Dmitry Mouromtsev[1]

[1] International Laboratory of Information Science and Semantic Technologies, ITMO University, St. Petersburg, Russia
gwohlg@itmo.ru

[2] Higher School of Economics, National Research University, Moscow, Russia

Abstract. In this research, we manually create high-quality datasets in the digital humanities domain for the evaluation of language models, specifically word embedding models. The first step comprises the creation of unigram and n-gram datasets for two fantasy novel book series for two task types each, analogy and doesn't-match. This is followed by the training of models on the two book series with various popular word embedding model types such as word2vec, GloVe, fastText, or LexVec. Finally, we evaluate the suitability of word embedding models for such specific relation extraction tasks in a situation of comparably small corpus sizes. In the evaluations, we also investigate and analyze particular aspects such as the impact of corpus term frequencies and task difficulty on accuracy. The datasets, and the underlying system and word embedding models are available on github and can be easily extended with new datasets and tasks, be used to reproduce the presented results, or be transferred to other domains.

Keywords: Word embeddings · Digital humanities dataset · Dataset evaluation

1 Introduction

Language technologies are increasingly adapted in the field of digital humanities, which is also reflected by a rising number of scientific events[1]. In this publication, we focus on the domain of literary texts, and analyze certain aspects of two well-known fantasy novel book series, namely "A Song of Ice and Fire" (ASOIF) by George R. R. Martin, and "Harry Potter" (HP) by Joanne K. Rowling.

The research question is how well current NLP methods, especially in the form of word embedding models, perform on extracting and verifying fine-grained

[1] For Example: https://www.clarin-d.net/en/current-issues/lt4dh.

relations and knowledge from the comparably small book series corpora. Firstly, we manually create high-quality datasets for the two novel book series, including various *analogy* and *doesn't-match* tasks (for the precise task and dataset descriptions see Sect. 3). In total the number of test units is 31362, separated into 80 task sections. The tasks include analogies of type *husband*::*wife*, *sigil-animal*::*house*, *geographic-entity-name*::*location-type*, and many others. Secondly, we train various types of unigram and n-gram word embedding models on the book corpora, and evaluate their performance on the test datasets. As a remark, it is not possible to apply existing generic embeddings, since the majority of entities are out of vocabulary in pre-trained models.

As most of the tasks are very hard for word embedding models trained on small corpora, we do not expect high numbers in accuracy, but rather aim to provide easily reproducible baselines for future work on the given test datasets. In order for other researchers to compare their methods on the test data, we make all datasets, word embeddings models and evaluation code available online[2]. By making the dataset creation and evaluation processes simple and transparent, we aim to provide the basis for the extension of the existing datasets, the addition of new datasets, and the enhancement of the evaluation code base.

To shed more light on the research question, in the evaluations we analyze specific aspects which influence model accuracy, such as the impact of term frequency in the book corpus on performance, or the impact of task difficulty.

The remainder of this paper is structured as follows: Sect. 2 introduces related work, in Sect. 3 we will describe the task types and the methods of dataset creation and dataset structure, as well as the implementation. Section 4 presents the evaluation setup and evaluation results for the two book series, and Sect. 5 concludes the paper.

2 Related Work

In the NLP field, word embeddings have become a very popular in recent years for language modeling and feature learning, especially since the work of Mikolov et al. [1] on the word2vec toolkit in 2013. Word2vec includes a component to evaluate the accuracy of its analogy feature, and an accompanying general-domain dataset. Those components inspired parts of the datasets in the fantasy novel domain presented in this publication. Other well-known word embedding types include GloVe [2], fastText [3] or LexVec [4] – which will be evaluated in Sect. 4. In order to achieve good predictive quality, word embedding models are usually trained on large corpora with billions of tokens. Here, we evaluate the applicability within a specialized domain and with a small corpus.

Ghanny et al. [5] compare different types of word embeddings, including the word2vec CBOW and skip-gram versions, GloVe, and word2vec-f (which was trained on the output of a dependency parser). Model performance depends on task and situation. The authors have best results with word2vec-f on some NLP tasks, GloVe for analogical reasoning, and word2vec for word similarity tasks.

[2] https://github.com/cicling2018-dhdata/dh-dataset.

Wohlgenannt et al. [6] present and evaluate methods to extract social networks using co-occurrence statistics and word embeddings from a novel book series. The proposed work extends previous research with specific relation types like analogical reasoning and doesn't-match, the manual creation and provision of datasets, and is applied to multiple fantasy book series.

We further relate our work to two trends in Natural Language Processing. First of all, fantasy books recently became a popular subject of study in the Digital Humanities field due to several factors: i) such books often have a linear timeline suitable for timeline and storyline extraction [7], b) the books feature a profound amount of direct speech for dialog [8] and social network analysis [9]. Secondly, word embeddings and their applications are also widely studied in various fields [10], e.g. for hyponymy detection [11], to track diachronic changes [12] or to find corpus specific senses [13].

3 Methods

This section includes a description of the task types of analogical reasoning and doesn't match, gives some details on the dataset creation process, and the word embedding types used. Finally, it provides a quick overview of the implementation and links to the online repository.

3.1 Task Types

In order to be compatible with existing tools we focus on two task types. Firstly, we created and evaluated a dataset for the analogical reasoning (analogy) task. The classical example from the original word2vec implementation is: `king - man` + `woman` = `?`, where the correct solution is `queen`. With embedding models this task can be simply solved with vector arithmetic. The word2vec toolkit contains a general domain dataset with tasks such as

$capital\_\text{city}_a$, $country_a :: capital\_city_b, ?$

or $adjective_a$, $superlative_a :: adjective_b, ?$.

Using vector arithmetic, a candidate is selected from the whole vocabulary of the model, making this task quite hard due to the huge number of candidates.

The second task type is the *doesnt_match* task according to the implementation within the popular Gensim library[3]. Here, from an input of n terms, the system has to decide which term does not belong to this list. This task is obviously much easier, as the system only has a small set of terms (e.g. four input terms) to choose from, and not the whole vocabulary.

3.2 Dataset Creation

In the course of this research we created two datasets each (analogical reasoning and doesn't_match) for a "A Song of Ice and Fire" (ASOIF) and "Harry Potter" (HP). With the addition of n-gram datasets, the result are eight datasets.

[3] https://radimrehurek.com/gensim.

Inspired from the data found on the book wikis[4,5] we collected test data, and manually filtered, edited and extended the data to ensure high quality. Two domain experts each worked on the datasets for the two book series. Manual refinement was necessary for a number of reasons: i) Many of the terms in the Wiki have a very low frequency in the book text, some do not appear at all (but only in the extensive book appendices of ASOIF or related books by the same author). ii) We removed ambiguous terms from the datasets, as currently the focus of the datasets is not word sense disambiguation, but relation modeling. As an example, in the ASOIF world, "Nymeria" is the name of Arya's direwolf, but also the first name of Nymeria Sand (a character). iii) Additionally to our original unigram models and datasets, to properly capture some of the entities, it was necessary to also provide n-gram models and datasets. The selection of proper surface forms for n-grams, and the assignment to the unigram or n-gram dataset also required some manual intervention.

3.3 Word Embedding Models

To address the tasks defined in the dataset, obviously, many techniques and combinations of methods are feasible. At the current stage, we focus on the application of word embedding models. Word embedding models have been shown to be very successful on many NLP tasks [5], and furthermore they are easy to train, apply and compare.

In the next section we evaluate various word embedding models based on these four well-known model types:

word2vec: Word2vec [1] was developed by a team of researchers at Google. It applies two-layer neural networks which are trained to reconstruct the linguistic context of words. In training, the input is a (typically large) corpus, the results are the word embeddings. Every word (above the threshold frequency) is assigned a dense vector of floating point values, usually in the range of 50–300 dimensions. Proximity in vector space reflects similar contexts in which words appear. word2vec includes two model architectures: CBOW and skip-gram. With CBOW, the model predicts the current word by using a window of surrounding words. Using skip-gram, the model predicts the surrounding context of the current word.

GloVe: GloVe [2] takes a different route to learning continuous vector representations of words. GloVe applies dimension-reduction on a word-word co-occurrence matrix. The model is trained to learn word vectors such that their dot product equals the logarithm of the words' probability of co-occurrence.

FastText: FastText [3] is based on the skip-gram model, but also makes use of word morphology information in the training process. By using sub-word information, fastText can also supply vectors for out-of-vocabulary words.

[4] http://awoiaf.westeros.org/index.php?title=Special:Categories.
[5] http://harrypotter.wikia.com/wiki/Main_Page.

LexVec: Finally, LexVec [4] uses a weighted factorization of the Positive Pointwise Mutual Information (PPMI) matrix via stochastic gradient descent. It employs high penalties for errors on frequent co-occurrences, and performs very well on word similarity and analogy in the experiments by the authors.

3.4 Implementation

The implementation has two main components, the creation of datasets and the use of those datasets to evaluate the models. As stated, all code, the word embedding models, and the datasets are available online[6]. The results presented here can be reproduced by cloning the repository and running the evaluation scripts, and all parts (datasets, models, etc.) can be extended easily.

Regarding the creation of datasets, the system uses a simple format to define the task units for any section in the dataset, e.g. all the *child::father* relations. The data format is documented in the github repository. From the definitions, the `create_questions.py` script creates the evaluation dataset as all permutations of the input definitions.

For the evaluation of the system, there are two main scripts for the *analogies* and *doesn't-match* tasks. Both first iterate over the word embedding models defined in the configuration file, run the task units from the datasets, and obviously collect and aggregate all the evaluation results. The implementation makes use of the Gensim library [14] for loading the models, and performing the two basic task types. For details on script usage and dataset extension see the system documentation online on github.

4 Evaluation

The evaluation section first describes specifics of the text corpora, the model settings and the datasets, and then provides the evaluation results for analogical reasoning and the doesn't-match tasks.

4.1 Evaluation Setup

Text Corpora. As already mentioned, the experiments were run on the plain text corpora of "A Song of Ice and Fire" (ASOIF) by George R. R. Martin (books 1–4), and "Harry Potter" (HP) by Joanne K. Rowling (all books). The ASOIF corpus has a size of 6.9M and contains about 1.3M word tokens. The HP series is of similar size, with 6.5M file size and 1.1M tokens. Corpus preprocessing only consisted of the removal of punctuation, quotation marks and newline characters. In order to support also datasets for n-grams, we applied the word2phrase tool from word2vec to create the n-gram corpus versions.

[6] https://github.com/cicling2018-dhdata/dh-dataset.

Models Trained. To compare and evaluate the performance of various word embedding types, we trained the following models on the two corpora (all available on github):

w2v-default: This is a word2vec model trained with the default settings: 200 vector dimensions, skip-gram, word window size:5, etc.[7].
w2v-ww12-300-ns: window size of 12, 300-dim., negative sampling.
w2v-CBOW: same settings as *w2v-ww12-300*, but CBOW instead skip-gram.
GloVe: defaults (window size 15). Except: 200-dim vectors instead 50-dim.
fastText: We used the default settings, except: 25 epochs, window size of 12
LexVec: We used the default settings, except: 25 epochs, window size of 12

The models and test dataset are available in versions for unigrams and n-grams. The n-gram data can be easily recognized by the suffix "`_ngram`" as part of the respective filenames. All models are available online in the github repository.

4.2 Dataset Description

The dataset creation process is described in Sect. 3.2. The resulting datasets are:

questions_soiaf_analogies.txt: Analogical reasoning tasks for the ASOIF series. Contains 8 different task types with a total of 2848 tasks.
hp_soiaf_analogies.txt: Analogical reasoning tasks for the Harry Potter series. 17 different task types and 4790 tasks in total.
questions_soiaf_doesnt_match.txt: Doesn't-match tasks for ASOIF, with 13 sections, and 11180 total tasks.
questions_hp_doesnt_match.txt: Doesn't-match tasks for the HP series, with 19 sections, and 8340 total tasks.
Additionally to these four datasets, there exist four more datasets for n-grams, with similar filenames, but including the marker "`_ngram`". The n-gram datasets include in total 4204 task units in 23 sections.

4.3 Analogy Task Results

As mentioned, in the analogy task, the input is a triple of terms, which can be read as x_1 is to x_2, what y_1 is to $y_?$. For example, *man* is to *king* what *woman* is to X. Or, in ASOIF, *kraken* is to *Greyjoy* what *lion* is to X' (correct: *Lannister*). The evaluated system has to guess the correct answer from the whole vocabulary. The vocabulary size in the models used here is between $11K$ to $60K$ terms. Given the comparably small corpus size, and the ambiguity of relations between terms, this task is very hard.

Every task is defined by four terms, three input terms, and the correct answer. The tasks are split into various sections, for example predicting *child-to-father* relations, *houses-to-their-seats*, etc. The ASOIF and HP datasets contain in total 7638 task units. For every unit, Gensim uses vector arithmetic to calculate the candidate term, and compares it to the correct solution given in the dataset.

[7] -cbow 0 -size 200 -window 5 -negative 0 -hs 1 -sample 1e-3 -threads 12.

Table 1. ASOIF analogy dataset: accuracy of various word embedding models on various selected analogy task types, and total accuracy.

Task type	First-name-last-name	Child-father	Husband-wife	Geo-name-location	Houses-seats	Total
Number of tasks:	2368	180	30	168	30	2848
w2v-default	20.73	3.33	6.67	1.19	43.33	18.43
w2v-ww12-300-ns	39.53	1.67	0.0	10.71	26.67	34.38
w2v-CBOW	0.46	6.11	10.0	7.14	6.67	1.37
GloVe	40.58	6.11	3.33	1.19	26.67	**34.8**
fastText	37.67	4.44	0.0	13.1	26.67	32.94
LexVec	40.08	3.89	6.67	0.0	23.33	34.38

Table 2. Harry Potter analogy dataset: accuracy of various word embedding models on various selected analogy task types, and total accuracy.

Task type	First-name-last-name	Child-father	Husband-wife	Name-species	Total
Number of tasks:	2390	224	72	566	4790
w2v-default	21.51	11.61	11.11	21.02	21.82
w2v-ww12-300-ns	50.46	11.16	43.06	32.69	34.11
w2v-CBOW	2.13	6.7	0.0	9.36	4.59
GloVe	43.68	6.7	22.22	25.27	30.96
LexVec	43.97	8.04	37.5	36.93	**34.72**
fastText	38.33	3.57	33.33	13.25	23.88

Table 1 presents the evaluation results for the ASOIF analogy dataset. Due to space limitations, we selected the results for five dataset sections, and the aggregated results. Best results were provided by *GloVe*, but there is no considerable difference to word2vec variants like *w2v-ww12-300-ns* and to *LexVec*. Both the *w2v-default* setting with its small word window of 5, and especially the word2vec *CBOW* method are unsuited for the task. Generally the accuracy is low with ca. 34%, reasons for the difficulty of the task setting are given above (Table 2).

The evaluation of analogical reasoning with the dataset for the HP books confirms the observations made on the ASOIF dataset. Here, the setting *LexVec* performs best, followed by *w2v-ww12-300-ns* and *GloVe*. Again, the score of *w2v-CBOW* is extremely low. The data suggests that for the analogy task with such a small corpus a large word window is helpful to counter data sparsity.

As a general remark, it was challenging to create a large number of high-quality relations for analogical reasoning. Partly because of ambiguity problems, as there are many re-occurring names and nicknames of entities in the large ASOIF universe. Furthermore, relations over change over time, for example in the ASOIF series, the character *Jon Snow* is first wrongly thought to be a bastard son of *Ned Stark*, which is later revealed as not true.

4.4 Doesn't_Match Task Results

In the doesn't-match task, the input dataset contains four terms, where three are semantically connected, and one intruder is mixed in. To distinguish the correct answer, it is explicitly provided in the dataset after a term separation symbol. The evaluation script calls Gensim to provide the intruder candidate, which is then compared to the correct answer. Internally, Gensim computes the mean vector from all input vectors, and then calculates the cosine distances. The vector with the largest distance is selected as not matching the rest. The random baseline for this task is $\frac{1}{4}$ (0.25).

Table 3. ASOIF doesnt_match dataset: accuracy of various word embedding models on selected doesn't_match task types, and total accuracy.

Task type	Family-siblings	Names-of-houses	Stark clan	Free cities	Total
Number of tasks:	160	7280	1120	700	11180
w2v-default	85.63	65.32	94.29	90.43	**74.54**
w2v-ww12-300-ns	85.63	53.7	88.39	91.86	66.85
w2v-CBOW	78.13	50.25	84.29	85.14	62.21
GloVe	80.63	67.69	89.64	88.71	73.28
fastText	84.38	54.3	84.64	91.14	66.86
LexVec	80.0	56.92	81.96	90.43	67.51

Table 3 gives an overview of the results on the 11180 task units defined for ASOIF. Due to the lower task complexity accuracy numbers are around 75%, with best results for *w2v-default* and *GloVe*. Both for ASOIF and for Harry Potter (Table 4) *w2v-default* performs surprisingly well on the doesn't-match tasks, which suggests that for this task type (semantic word similarity) a smaller and more focused word context has benefits.

Table 4. HP doesnt_match dataset: accuracy of various word embedding models on selected doesn't_match task types, and total accuracy.

Task type	Family-members	Gryffindor-members	Magic-creatures	Wizards-Animagi	Total
Number of tasks:	440	2800	700	200	8340
w2v-default	85.23	82.07	59.86	72.0	**71.16**
w2v-ww12-300-ns	88.18	73.18	33.57	56.0	66.38
w2v-CBOW	56.59	53.36	40.71	52.0	55.78
GloVe	84.09	69.46	35.57	53.0	65.49
fastText	85.68	76.71	35.14	34.5	64.84
LexVec	83.86	69.5	39.71	60.0	61.92

The Harry Potter dataset defines a total of 8340 task units in 19 sections. The evaluation results are in line with the ASOIF evaluations, with the best results for *w2v-default* with an accuracy of 71.16%, followed by other word2vec variants, GloVe, and fastText. Again, results per task section vary considerably. Section 4.6 analyzes the influence of term frequencies in the corpus on accuracy, and Sect. 4.7 discusses the impact of task difficulty. Our results also confirm Ghanny et al. [5], with good results for GloVe on analogical reasoning, and word2vec for word similarity.

4.5 Results for N-Gram Datasets and Models

Additionally to the unigram data, we also trained n-gram models and created n-gram datasets. For the ASOIF book series, we define 192 task units for analogical reasoning and 2000 units for the doesn't-match task, the respective numbers for HP are 92 and 1920. As the results of the n-gram evaluations are in many aspects in line with unigram data, we will keep the analysis short and focus on differences to the unigram results. Performance on analogical reasoning is generally lower, with only about 10–15% accuracy in the n-gram setting. This might (in part) be caused by lower term frequencies associated with n-grams. For the doesn't-match task, accuracy is similar to the unigram evaluation, with the exception of fastText embeddings. As fastText uses subword information, it performs very well on n-grams which share lexical elements. For the ASOIF dataset, fastText provides 92.1 accuracy for the doesn't-match task.

4.6 Impact of Term Frequencies

We investigate the correlation between the frequency of the terms in the respective book corpus, and the correct guesses in the doesnt-match task. In general, the expectation was that a higher frequency of terms in the book leads to a "better" word embeddings vector for the term, and therefore to higher results. Our experiments confirm this expectation only in part. As every task unit consists of four terms (the three that are related, and the intruder to be found), the question is which of the term frequencies to use. Table 5 shows the *correlation scores* for different aspects of term frequency: (i) the frequency of the real *intruder* which had to be found, (ii) the frequency of the term chosen as intruder using the model, (iii) the frequency bin of the chosen intruder, and (iv) the average frequency of all four terms in the task unit.

The data suggests that the intruder is not easier to find if it has a higher frequency in the books (column "frequency of real intruder" in Table 5). One reason might be that very frequent entities often change their context as the story progresses. E.g., in ASOIF the *Arya* character first lives as child with her family, then moves into the context of the capital city, later travels the country with changing companions, and finally ends up to be trained as assassin on a different continent, so the context changes are drastic. More research is needed to investigate the phenomenon of little influence of the frequency of the intruder.

On the other hand, the frequency of the intruder guessed using the model is correlated with accuracy, the level of correlation is similar to the correlation between task difficulty and accuracy. To further analyze this finding, we created

Table 5. Correlations of term frequencies with accuracy in tasks for unigram and n-gram models trained on *Harry Potter* and *A Song of Ice and Fire*

Correlation of accuracy to		Freq. of real intruder	Freq. of chosen intruder	Freq. bin of chosen intr.	Avg. freq. of task terms	Difficulty
ASOIF	Unigram	−0.10	0.31	0.29	0.05	0.30
	N-gram	0.15	0.17	0.18	0.12	0.10
HP	Unigram	−0.03	0.05	0.20	0.02	0.18
	N-gram	−0.25	0.31	0.58	−0.19	0.25
Average results		−0.06	0.21	0.31	0.00	0.21

frequency bins and measured the accuracy per bin, see Table 6 for results for the LexVec models.

Table 6. Accuracy of *doesn't match* tasks depending on the frequency of the term suggested by the word embedding model. Evaluated using LexVec embeddings.

Correct results		Bin 1	Bin 2	Bin 3	Bin4	Bin 5
ASOIF	Unigram	0.29 (140)	0.82 (694)	0.40 (2031)	0.63 (5262)	0.92 (3053)
	N-gram	0.61 (431)	0.57 (349)	0.84 (268)	0.85 (420)	1.00 (532)
HP	Unigram	0.32 (800)	0.64 (1033)	0.47 (1776)	0.71 (3293)	0.73 (1438)
	N-gram	0.13 (676)	0.19 (321)	0.28 (479)	0.89 (131)	0.97 (313)
Average results		0.337	0.555	0.525	0.77	0.905

Table 6 presents the evaluation results depending on the frequency bin of the intruder suggested by the model. Terms are categorized as follows: Bin 1 contains the terms occurring in the books less than 20 times, bin 2 is for terms with 20–50 occurrences, bin 3 for frequency 50–125, bin 4 for frequency 125–500, and bin 5 for term frequencies above 500. The table shows the ratio of correct guesses in the *doesn't match task* for the HP and ASOIF book series, with the number of tasks units per bin given in parenthesis. The general tendency is that term frequency of the suggested intruder impacts accuracy, but the evaluations show high variability.

4.7 Impact of Task Difficulty

Finally, we analyze the impact of task difficulty on accuracy. In the *doesn't match*-task, task difficulty, can be controlled via the level of similarity of the mixed-in term to the other three terms. To this end, in the datasets we define 20 intruding terms per doesn't match triple, which are grouped into four difficulty categories (1–4), with five terms per category. In the hardest category mixed-in terms are semantically very close and of same syntactic type or named entity class, whereas in the easiest category, there is no semantic relation of the intruder to the doesn't match triple. In the ASOIF dataset for example, in section "family-siblings" for the triple of siblings *Jaime Tyrion Cersei*, in the hardest category, intruding terms are e.g. *Tywin* (father) or *Joffrey* (son of the siblings). And in the easiest category, an example of an intruding term is *dragon* (not related).

Table 7. Accuracy of ASOIF and HP datasets in *doesn't-match* tasks depending on task difficulty.

Datasets/Models	Task difficulty				
	1 (hard)	2 (rather hard)	3 (rather easy)	4 (easy)	Total
ASOIF GloVe	54.28	63.32	86.26	**89.27**	74.54
HP LexVec	47.62	68.29	81.91	**86.81**	71.16
ASOIF Ngram fastText	67.00	74.60	85.00	**93.60**	80.05
HP Ngram w2v-default	48.54	72.01	87.08	**95.83**	75.89

Table 7 presents the results of the doesn't-match tasks based on task difficulty levels, showing the results for unigram and n-gram datasets of selected embedding models. The impact of task difficulty on accuracy is clearly evident from the data, for the hardest task category accuracy is around 50%, which leaves a lot of room for future work.

5 Conclusion

In this publication, we introduce datasets in the digital humanities domain for evaluating word embedding and relation extraction systems, and apply various embedding methods to provide benchmark evaluations. The datasets include 31362 analogical reasoning and doesn't-match relations for the "A Song of Ice and Fire" (by G.R.R. Martin) and "Harry Potter" (by J.K. Rowling) book series.

The contributions of this work are as follows: (i) the manual creation of high-quality datasets in the digital humanities domain, (ii) providing models for the two book series trained with various word embedding techniques, (iii) evaluating the suitability of word embeddings trained on small corpora for the tasks at hand, including the evaluation of unigram and n-gram tasks, and analyzing the impact of task difficulty and corpus term frequencies on accuracy, and finally (iv) the provision of the code-base and related resources, which are easy to extend with new task types, test data, and models.

A very interesting direction of future research is leveraging multimodal data (images, videos, etc.) available in abundance in the given domain for the creation of entity representations. Thoma et al. [15] propose multimodal embeddings which combine embeddings learned from text, images, and knowledge graphs. Furthermore, we plan to apply crowdsourcing on a subset of task units to get an estimate of human level accuracy on the various task types and difficulty levels.

Acknowledgements. This work was supported by the Government of the Russian Federation (Grant 074-U01) through the ITMO Fellowship and Professorship Program. Furthermore, the article was prepared within the framework of the Basic Research Program at the National Research University Higher School of Economics (HSE) and supported within the framework of a subsidy by the Russian Academic Excellence Project '5-100'. It was supported by the RFBR grants 16-29-12982, 16-01-00583.

References

1. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
2. Pennington, J., Socher, R., Manning, C.D.: Glove: global vectors for word representation. In: EMNLP, pp. 1532–1543 (2014)
3. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. arXiv preprint arXiv:1607.04606 (2016)
4. Salle, A., Idiart, M., Villavicencio, A.: Enhancing the LexVec distributed word representation model using positional contexts and external memory. CoRR abs/1606.01283 (2016)
5. Ghannay, S., Favre, B., Esteve, Y., Camelin, N.: Word embedding evaluation and combination. In: Calzolari, N., et al. (eds.) LREC 2016. ELRA (2016)
6. Wohlgenannt, G., Chernyak, E., Ilvovsky, D.: Extracting social networks from literary text with word embedding tools. In: Proceedings of Workshop LT4DH, Osaka, Japan, COLING 2016 (2016) 18–25
7. Laparra, E., Aldabe, I., Rigau, G.: From timelines to storylines: a preliminary proposal for evaluating narratives. In: Proceedings of the First Workshop on Computing News Storylines, pp. 50–55 (2015)
8. Flekova, L., Gurevych, I.: Personality profiling of fictional characters using sense-level links between lexical resources. In: EMNLP, pp. 1805–1816 (2015)
9. Bonato, A., D'Angelo, D.R., Elenberg, E.R., Gleich, D.F., Hou, Y.: Mining and modeling character networks. In: Bonato, A., Graham, F.C., Prałat, P. (eds.) WAW 2016. LNCS, vol. 10088, pp. 100–114. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-49787-7_9
10. Hellrich, J., Hahn, U.: Bad company-neighborhoods in neural embedding spaces considered harmful. In: COLING, pp. 2785–2796 (2016)
11. Ustalov, D., Arefyev, N., Biemann, C., Panchenko, A.: Negative sampling improves hypernymy extraction based on projection learning. In: EACL 2017, p. 543 (2017)
12. Hamilton, W.L., Leskovec, J., Jurafsky, D.: Diachronic word embeddings reveal statistical laws of semantic change. arXiv preprint arXiv:1605.09096 (2016)
13. Kågeback, M., Johansson, F., Johansson, R., Dubhashi, D.: Neural context embeddings for automatic discovery of word senses. In: Proceedings of 1st Workshop on Vector Space Modeling for NLP, pp. 25–32 (2015)

14. Řehůřek, R., Sojka, P.: Software framework for topic modelling with large corpora. In: Proceedings of the LREC 2010, Valletta, Malta, ELRA, pp. 45–50 (2010)
15. Thoma, S., Rettinger, A., Both, F.: Towards holistic concept representations: embedding relational knowledge, visual attributes, and distributional word semantics. In: d'Amato, C., et al. (eds.) ISWC 2017. LNCS, vol. 10587, pp. 694–710. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68288-4_41

One World - Seven Thousand Languages (Best Paper Award, Third Place)

Fausto Giunchiglia(✉), Khuyagbaatar Batsuren, and Abed Alhakim Freihat

University of Trento, 38100 Trento, TN, Italy
fausto.giunchiglia@unitn.it

Abstract. We present a large scale multilingual lexical resource, the *Universal Knowledge Core* (UKC), which is organized like a *Wordnet* with, however, a major design difference. In the UKC the meaning of words is represented not only with *synsets* but also using language independent *concepts* which cluster together the synsets which, in different languages, codify the same meaning. In the UKC, it is concepts and not synsets, as it is the case in the Wordnets, which are connected in a semantic network. The use of language independent concepts allows for the native integrability, analysis and use of any number of languages, with important applications in, e.g., multilingual language processing, reasoning (as needed, for instance, in data and knowledge integration) and image understanding.

Keywords: Multiligual lexical · Semantic networks · Multilingual language processing

1 Introduction

Since the seminal work on the *Princeton WordNet* (PWN) [1] a lot of effort has been devoted to the production of large scale lexical resources, some of which are multilingual, see, e.g., [2–4].[1] These resources, also because largely constructed by "replicating" the PWN, share its basic organizational principles and, in particular, the fact that the multiple meanings of a *word* are codified as a *lexicalized concept*, also called a *synset*, consisting of a (possibly incomplete) set of synonymous words. Furthermore, in a multilingual Wordnet, each language is developed as if it was the only language and then, if s_1 is the synset of a word w_1 in a reference language L_1, usually PWN English, then the synset s_2 of the corresponding word w_2 in L_2 (i.e., the translation of w_1 in L_2) is linked to s_1.

Our goal in this paper is to describe a multilingual lexical resource that we call the *Universal Knowledge Core (UKC)*.[2] The UKC shares all the PWN design choices but one: the *synsets* which in different languages codify the same meaning

[1] See http://globalwordnet.org/ for a compilation of the most relevant resources available today.

[2] The word *knowledge* in *UKC* is motivated by our focus on studying language not *per se* but as a key component of reasoning systems.

A. Gelbukh (Ed.): CICLing 2018, LNCS 13396, pp. 220–235, 2023.
https://doi.org/10.1007/978-3-031-23793-5_19

Table 1. Language distribution.

#Words	#Languages	Samples
>90000	2	English, Finnish
>75000	4	Mandarin, Japanese, etc.
>50000	6	Thai, Polish, etc.
>25000	17	Portuguese, Slovak, etc.
>10000	29	Islandic, Arabic, etc.
>5000	39	Swedish, Korean, etc.
>1000	66	Hindi, Vietnam, etc.
>500	85	Kazakh, Mongolian, etc.
>0	335	Ewe, Abkhaz, etc.

are clustered into *language agnostic concepts*. Furthermore, in the UKC, semantic relations link concepts, and not synsets, and create a *language independent semantic network*, that we call the *Concept Core (CC)*. So far, the UKC has evolved as a combination of importing of freely available resources, e.g., WordNets or dictionaries of high quality, and language development, see e.g., [5]. As of to day, it contains 335 languages, 1,333,869 words, 2,066,843 synsets and more than 120,000 concepts. Table 1 reports the distribution of words over languages where, more or less, 90% of the words belong to 50 languages.[3]

The existence of the CC makes the UKC *not biased by any language and culture* and, therefore, *inherently open* and easily extensible. For instance, *lexical gaps*, namely previously missing concepts lexicalized in a new language can be dealt with by adding a new concept, thus solving one of the difficulties which arise in the construction of multilingual Wordnets. This is crucial given that the languages of the so called WEIRD (Western, Educated, Industrial, Rich, Democratic) cultures cannot in any way be taken as paradigmatic of the world languages [6]. Furthermore, it is also important to notice how the co-existence of synsets and concepts allows for the seamless integration of language dependent and language independent reasoning. Thus, on one side, any application using concepts will automatically run for any language supported by the UKC, see, e.g., the work on cross-lingual data integration described in [7], while, on the other side, as discussed in detail in Sect. 3, synsets can be used to keep track of the local language and culture. An exemplary application is the extension to multiple languages for the work in [8,9] which uses Wordnet for the large scale classification of photos (what is depicted by a photo is biased by culture; compare, e.g., the photo of a home in Italy with that of a home in Mongolia).

The paper is organized as follows. In Sect. 2 we describe the organization of the UKC. In Sect. 3 we describe the complementary roles of words, synsets and concepts. In Sect. 4 we describe the resulting three layer organization of the UKC. In Sect. 5 we define the notion of language diversity. In Sect. 6 we deal with the issue of the quality of the resources and of the UKC in particular. The related work and our plans for the future work conclude the paper.

[3] From February 2018, the UKC will be browsable on line at the link http://kidf.eu.

2 The Language and the Concept Core

The key design principle underlying the UKC is to maintain a clear distinction between the *language(s)* used to describe *the world as it is perceived* and what is being described, i.e., the world itself. The *Concept Core (CC)* is the UKC representation of the world and it consists of a semantic network where the nodes are language independent *concepts*. Each concept is characterized by a unique identifier which distinguishes it from any other concept. The semantic network consists of a set of semantic relations between nodes which relate the meanings of concepts, where these relations are an extension of those used by the PWN (e.g., *hyponym*, *meronym*).

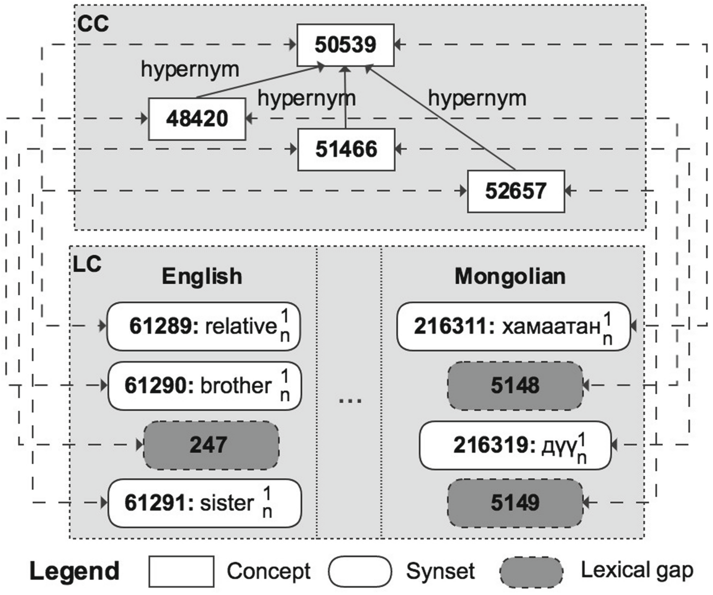


Fig. 1. A fragment of the semantic network of concepts and their synsets.

We talk of the *Language Core (LC)*, meaning the component that, in the UKC, corresponds to the PWN, namely the set of *words*, *senses*, *synsets*, *glosses* and *examples* supported by the UKC. Despite playing a similar role, the LC is actually quite different from the PWN. Similarly to the PWN, in the LC each synset is univocally associated with one language and, within that language, with at least one word. Differently from the PWN, synsets are linked to concepts, and there is the constraint that each synset is linked to one and only one concept. There is, furthermore, the constraint that, *for a concept to be created, there must be at least one language where it is lexicalized.* Given the multilinguality of the UKC, there is a one-to-many relation between concepts and synsets. Figure 1 shows how synsets and concepts are related ("n" means that the reference word is a noun, "1" that synset is associated to its first sense).

Glosses and *examples* are associated with synsets, as in the PWN. We have evaluated the possibility of associating glosses also to concepts. Ultimately, we decided that this should not be the case as such a description would be linguistic in nature and there is no universal language which could be used to describe all the concepts in the CC. One difference with the PWN is that, in the UKC, lexical gaps have glosses, even if they do not have examples (which would be impossible). The intuition is that the gloss of a lexical gap can be seen as "local" language dependent description of a missing synset. This choice has turned out to be pragmatically useful when one is interested in understanding the meaning of a lexical gap without knowing the language(s) which generate(s) them.

3 Words, Synsets and Concepts

Humans build representations of what they perceive, what we usually call *the world*, as complex combinations of *concepts* where, following [10], we take *concepts* to be *mental representations of what is perceived*. The recognition of a concept is taken to be the result of (multiple) *encounters*, i.e., events, $e_1, ..., e_n$, during which *substances manifest* themselves to a perceiver (e.g., an observer or a listener), where substances have two fundamental properties:

1. *they maintain some level of, but not full, invariance on how they manifest themselves to observers across multiple encounters* and
2. *this ability is an intrinsic property of substances.*

Examples of concepts generated from substances are objects (e.g., persons, cars, cats), actions (e.g., walk, drive) roles (e.g., father, president); see [10,11] for a detailed discussion about these notions and also [12] for the early work in the field of *Biosemantics* which introduced the notions of substance and encounter. The *key* observation is that we take concepts as representations denoting *sets of encounters*, rather than *sets of instances* which share a set of properties, as it is the case in the *Descriptionistic* theories of meaning, e.g., *Knowledge Representation* or the "usual" *logical semantics*. Thus, for instance, the denotation of the concept *car* is the set of times a car has been perceived, e.g., seen by me, rather than the set of cars which, e.g., are in Trento. This shift allows us to treat concepts and words uniformly. We take words, like concepts, to be representations of the world; more specifically, to be mental representations of mental representations of the world (i.e., of *concepts*). As such, words, like concepts, are the results of sets of encounters $e_1, ..., e_n$ during which they are perceived by, e.g., a listener of reader, as produced by, e.g., a human speaker or written text. Thus, for instance, analogously to what happens for the *concept car*, the *word car* denotes the set of input occurrences which are generated by looking at a set of documents and/or by hearing a set of utterances.

We represent words, synsets and concepts and their respective roles as in Fig. 2. Outside the UKC there is the world as we perceive it, e.g., via vision (bottom) or listening (top). At the bottom there are concepts $c_1, ..., c_n$, while, at

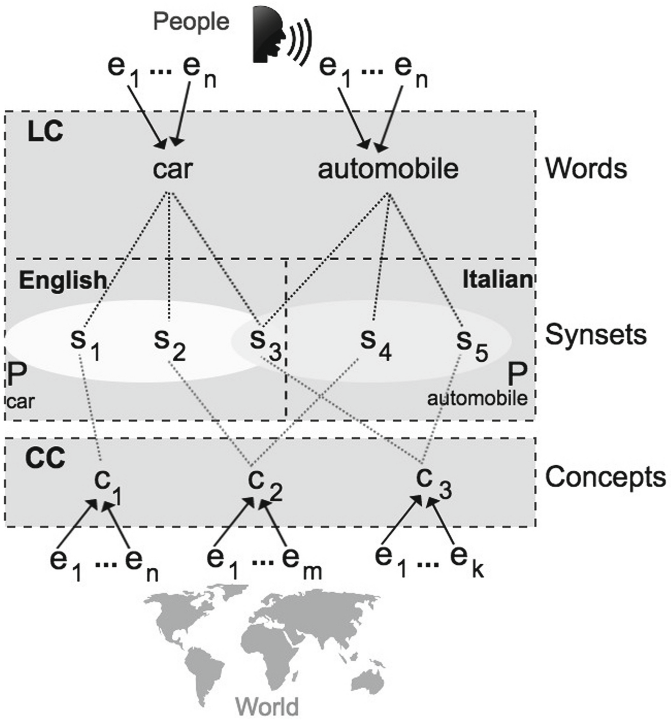


Fig. 2. The UKC and the world.

the top, there are words $w_1, ..., w_n$ (in Fig. 2, *car* and *automobile*), where both words and concepts are perceived as the result of the encounters $e_1, ..., e_n$.

Moving to the center of Fig. 2, the synsets $s_1, ..., s_n$ are linked to words and to concepts, see, e.g., the word *car* in Fig. 2. We call these two links *word sense* and *concept sense*, respectively, or simply *sense*, when the context makes clear what we mean. Notice how, as represented in Fig. 2, both words and concepts are ambiguous representations of synsets, in the sense that there is a one-to-many relation between them and synsets. The sense of a word depends on the *context* within which it is perceived while the sense of a concept depends on the *language* used. Thus, as in Fig. 2, the word *car* and the word *automobile* denote the sets of synsets P_{car} and $P_{automobile}$, respectively, where each synset is indexed by a different context, and these two sets overlap in s_3. In turn, the concept c_3, like any other concept, is denoted by a set of synsets, each synset belonging to a different language (English and Italian in Fig. 2). c_1, being non being lexicalized in Italian, is a probe for a possible lexical gap in this language. Notice also how there are words, e.g., *automobile* which are shared across languages, this being pervasive with languages with common roots, e.g., Portuguese and Brasilian Portuguese.

As a result the UKC implements the following stratified theory of meaning:

– the results of perception, i.e., words and concepts, denote the set of events during which they are perceived; they define the boundary between the UKC and the world;
– words denote sets of synsets, one per context of use;
– synsets denote concepts, where any concept is denoted by multiple synsets, one per language;
– Any triple $\langle w_i, s_i; c_i \rangle$, with s_i word sense of w_i and c_i concept sense of s_i, is a *Causal connection* $CC(w_i, c_i)$ between w_i and c_i.

$CC(w_i, c_i)$ implements the *causal connection between words and concepts* that humans exploit in knowledge representation and reasoning. Given that media, e.g., photos and videos, are direct representations of concepts, the above organization paves the way to integrated multimedia and multilanguage systems, extending the work in the integration of linguistic resources and media, so far done only for single languages, see, e.g., [8,9].

4 World, Languages and Model(s)

The three-layer organization of meaning into words, synsets and concepts, as represented in Fig. 2, motivates a three layer design of the UKC, as represented in Fig. 3, with the first two layers inside the LC and the third inside the CC. We have:

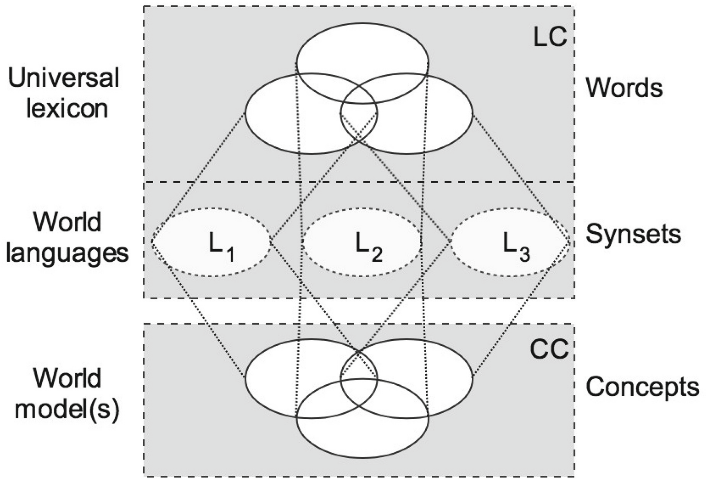


Fig. 3. Languages, universal lexicon and world model(s).

1. the *Word Layer*, which stores what we call the *Universal Lexicon*,
2. the *Synset Layer*, which stores the *World Languages*, and
3. the *Concept Layer*, which stores the *World (mental) model(s)*, as represented by the CC.

Word Layer and *Concept Layer* store the results of perception while the *Synset Layer* implements the causal connection between words and concepts. In the *Synset Layer* each circle represents a different language where all languages are mutually disjoint, this being a consequence of the fact that, differently from what is the case for words (see Fig. 2), each synset is associated with one and only one language. On this basis, in the UKC, we formally define *a Language as a set of synsets*, in formulas

$$L = \{s_i\}_{i \in I_L}.$$

The above definition is at the basis of all the definitions regarding language diversity and resource quality provided in the next sections. It allows, for instance, to compare the concepts lexicalized in the different languages, including the absence of lexicalizations (which are probes for lexical gaps) and to study how polisemy and synonymy map to the underlying concept semantic network.

The *Word Layer* stores the *Universal Lexicon*, namely the set of the words belonging to at least one language. Notice that a word, meant as the event by which it is perceived and recognized, does not a priori belong to any language. It is only a *sign* or a *sound* which may be used in more than on language and which is recognized as belonging to a language as part of the word sense disambiguation process. Of course, as represented in Fig. 3, it is possible to reconstruct the set of words of a Language from synsets using the inverse of the word sense relation.

The *Concept Layer* is a language agnostic representation of the world as we perceive it. *But, a model generated by who?* In the UKC, the world, as we perceive it, is taken to be a source of *perception events*. By perception event we mean here the concrete sensing action, performed by a sensing subject, which generates concepts and words, and the causal relations linking them. This gives us the possibility to define the notion of world (as we perceive it) in terms of the subject(s) which actually perform the sensing actions enabling the perception events.

According to a first mainstream interpretation, the CC is the model of the entire world, as it is generated by all the people (speaking all the languages) in the world. However, according to a second interpretation, the CC can also be seen as the union of the models of the world as they are generated by the different people (speaking the different languages) in the world, as represented in Fig. 3, e.g., the models of the 7,097 languages registered by the *Ethnologue project.*[4] Clearly these models intersect and are a subset (a subgraph) of the overall CC. It is interesting to notice how this view can be easily pushed to the extreme by associating a different world model to any different sensing subject (e.g., any person). During the generation of a lexicon, lexicographers would choose from a "common pot" words and concepts, namely what we all share via perception,

[4] http://www.ethnologue.com/.

while, at the same time, they would be able to decide synsets and senses, namely the causal relation from words to objects that is unique to each of us.

In this perspective, notice how *lexical gaps* are core to our studies on language diversity as they provide evidence of the different worlds perceived by people speaking different languages. The notion of lexical gap is seemingly quite intuitive: a lexical gap is a missing element in the lexicon. *But, missing with respect to what?* The approaches that we are aware of define this notion in terms of properties of the lexicon. Thus for instance, [13] defines lexical gaps as holes in the systematicity of languages while [14,15] define them as the lack of lexicalization detected when comparing two languages. [16] defines a lexical gap as a missing lexicalization of the semantic structure of a language, based on the analysis of the lexicon of that language. Our notion of lexical gap codifies directly the fact that a lexical gap is a missing link between a lexicon and semantics: *a lexical gap is a concept for which a language is known not to have a synset.* Notice how this definition relates directly to how different cultures generate, via language, different world models.

5 Language Diversity

The *diversity* of languages appears at many different levels, e.g., phonology, morphology, syntax, and has been the object of extensive studies in the field of *Historical Linguistics* [17] as well as the related field of *Linguistic Typology* [18]. Diversity has many causes, for instance, genetic ancestry (languages which derive from a common language will tend to share more language elements), geographic closeness (people living closer will tend to use many similar or identical words), similarity in culture (people with similar cultures will tend to model the world with similar languages), and so on.

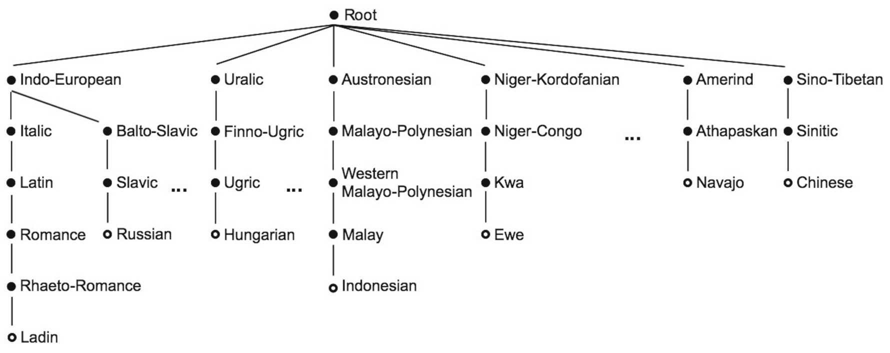


Fig. 4. The phylogenetic tree of languages.

We associate the notion of diversity to a set of languages $\mathcal{L}$, where the intuition is that the more different the languages, the higher the diversity measure.

Thus for instance the diversity measure of {Italian, French} will be smaller than that of {Italian, French, Spanish} and of {Italian, Mongolian}. We take the diversity of the empty set and of the singleton set to be zero. Furthermore, we talk of *relative diversity* (between two languages) when the cardinality of $\mathcal{L}$ is two. Finally, we talk of *language similarity* (and *relative similarity*) intuitively meaning the opposite phenomenon.

In this paper we concentrate on genetic diversity and adopt the notion introduced in [5] which, in turn, is an evolution of the measure defined in [19]. This notion is based on an analysis of the *Phylogenetic Tree* which describes how languages have progressively descended from other languages [19,20]. A fragment of this tree is depicted in Fig. 4 while Table 2 reports the UKC distribution of languages and continents over phyla (first nine columns), where around 60% of the languages belong to the first four phyla. In Fig. 4, the root is just a placeholder for the set of all languages, the intermediate nodes are *language families* or *phyla*, each associated with a set of languages, while the terminal nodes are the actual languages. There are nine children of the root which progressively split to consider all languages. We compute diversity based on the following intuitions:

1. The diversity measure of a language is its distance from the root node. For instance, as from Fig. 4, Russian is at distance 4 from the root while Ladin is at distance 6.
2. Higher nodes correspond to languages which split before and have evolved independently for more time, thus becoming more diversified. As from [5], this is modeled by associating to each node a weight λ^{-d} with $\lambda > 1$ and d being the distance from the root.
3. The diversity of a set of languages, when the set contains at least two languages, is the sum of the diversity of its languages.

The resulting diversity is not normalized. We normalize it by considering the diversity measure of a reference set, e.g., the languages in the Ethnologue project or, as we have done so far, with the diversity measure of the set $\mathcal{L}_{UKC}$ of the languages of the UKC. We call the resulting two measures *Absolute Diversity* and *(Normalized) Diversity* and we write them as AbsGenDiv and GenDiv, respectively.

Let us consider some examples, computed assuming $\lambda = 2$. The Absolute Diversity and Diversity of the languages of the UKC are AbsGenDiv($\mathcal{L}_{UKC}$) = 88.127 and GenDiv($\mathcal{L}_{UKC}$) = 1, respectively. Furthermore we have, as an example, AbsGenDiv({Hungarian, Italian, Polish, Russia, Basque}) = 3.469 and GenDiv({Hungarian, Italian, Polish, Russia, Basque}) = 0.039 (with respect to the $\mathcal{L}_{UKC}$).

6 Resource Quality

The languages in the UKC are far from being *complete*, i.e., from containing all the words and synsets used in the everyday spoken or written interactions, and far from being *correct*, i.e., from containing only correct senses, namely,

Table 2. Language distributions across phyla.

Phylum	Dep.	Lan.	EU	AS	AM	AF	PA	Example	LanInc(l)	LanQua(l)	#Words
Indo-European	7	115	86	26	1	1	1	English	[0.00; 1.00]	[0.00; 1.00]	[5; 147, 263]
Austronesian	6	36	1	23	2	0	10	Malay	[0.71; 1.00]	[0.16; 0.57]	[8; 24, 081]
Altaic	6	30	16	14	0	0	0	Mongolia	[0.53; 1.00]	[0.25; 0.62]	[12; 86, 574]
Uralic	6	22	22	00	0	0	0	Finnish	[0.01; 1.00]	[0.18; 0.57]	[8; 115, 259]
Niger-Kordofa	5	21	0	0	0	21	0	Zulu	[0.99; 1.00]	[0.24; 0.60]	[11; 354]
Amerind	4	18	0	0	18	0	0	Navajo	[0.98; 1.00]	[0.18; 0.59]	[10; 1, 460]
Sino-Tibetan	4	18	0	18	0	0	0	Mandarin	[0.10; 1.00]	[0.12; 0.65]	[3; 91, 898]
Afroasiatic	4	14	1	3	0	10	0	Hebrew	[0.91; 1.00]	[0.23; 0.61]	[15; 13, 601]
Caucasian	3	12	9	3	0	0	0	Chechen	[0.97; 1.00]	[0.22; 0.57]	[10; 2, 828]
Creole	3	9	0	0	5	1	3	Tok Pisin	[0.99; 1.00]	[0.22; 0.56]	[9; 485]
Small 22 families	4	40	4	11	17	4	4	Basque	[0.94; 1.00]	[0.26; 0.60]	[12; 25, 676]
Total	7	335	139	98	43	37	18	-	[0.00; 1.00]	[0.00; 1.00]	[0; 147, 263]

Dep. represents a depth of its corresponding phylum.
Lan. represents a number of languages existed in its corresponding phylum.
EU, **AS**, **AM**, **AF**, **PA** stand for continents, namely: Europe, Asia, Americas, Africa, and Pacific.
Note: each phylum of small families has no more than 5 languages.

Table 3. Ten sample languages from Table 2.

Language	ISO	#PsyMis	AvgDis	LanInc	LanQua
English	eng	14	3.42	0.00	1.00
Malay	msa	4,304	1.46	0.71	0.16
Mongolia	mon	6	1.16	0.99	0.50
Finnish	fin	7,471	1.22	0.01	0.27
Ewe	ewe	0	0	0.99	0.59
Navajo	nav	54	1.44	0.98	0.37
Mandarin	zho	2,596	1.17	0.09	0.38
Hebrew	heb	49	1.23	0.33	0.43
Chechen	che	0	0	0.99	0.61
Tok Pisin	tpi	22	1.68	0.99	0.28

only correct associations from words and concepts to synsets. This situation is unavoidable. No matter how developed a language is, it will always miss a lot of words and it will always embody the misconceptions, bias and also mistakes of the people who have developed it. As mentioned in the introduction, in the area of historical linguistics, the solution so far has been that of using small high quality resources; see for instance the work in [21], in lexicostatistics [22,23], mass comparison [24], or the recent work on lexical semantics described in [25]. However this approach seems even more problematic as it does not give anyhow a full guarantee of unbiasedness, it tends to crystallize the field on a small set of case studies and, because of this, it makes it hard to study the diversity of languages *at large*, which seems to be a long tail phenomenon.

As from [5], our approach is to define a set of *quantitative measures* and use them to evaluate the quality of a language and of the bias it introduces. For lack of space, we provide below a measure of incompleteness and one of incorrectness and exploit them to characterize some aspects of the current state of the UKC. A more complete list of measures will be provided in a follow up longer paper.

6.1 Incompleteness

The proposed notion of *Language Incompleteness* LanInc, with its dual notion of *Language Coverage* LanCov, is the direct extension of the notion of incompleteness of logical languages and theories. The idea is to exploit the fact that the CC can be taken as (a computational representation) the domain of interpretation of a language, defined as a set of synsets, and to count how much of it is not lexicalized by that language.

$$\mathrm{AbsLanCov}(l) = |\mathrm{Concepts}(l)| \tag{1}$$

$$\mathrm{LanCov}(l) = \frac{|\mathrm{AbsLanCov}(l)|}{|\mathrm{Concepts}(\mathrm{UKC})| - |\mathrm{Gaps}(l)|} \tag{2}$$

$$\mathrm{LanInc}(l) = 1 - \mathrm{LanCov}(l) \tag{3}$$

where Concepts(l) is the set of concepts lexicalized by a language l, Concepts(UKC) are the concepts in the UKC and Gaps(l) are the lexical gaps of l. AbsLanCov is the *Absolute Language Coverage*. Table 2 (column 10), reports the range of values for LangInc in the various phyla, while Table 3 provides its values for ten selected languages. It is interesting to notice how LangInc(English) = 0.0. This is indirect evidence of the English bias present in the current linguistic resources. It is a consequence of the fact that most Wordnets have been derived by PWN and that, so far, the UKC contains only concepts lexicalized in the PWN. The second observation is that all the languages not spoken by WEIRD societies are highly under-developed, for instance we have LangInc(Navajo) = 0.98.

6.2 Incorrectness

The quality of a language can be measured by several factors, e.g., translation mistakes, wrong senses, and much more. In the following, we analyze the problem of the *psycholinguistic mistakes* which we define as failures of adhering to the principle which, as from [26], states that *"... superordinate nouns can serve as anaphors referring back to their hyponyms. For example, in such constructions as 'He owned a rifle, but the gun had not been fired', it is immediately understood that the gun is an anaphoric noun with a rifle as its antecedent."* Fig. 5 provides an example of psycholinguistic mistake in the Spanish WordNet.

We have the following definitions:

$$\mathrm{AbsLanQua}(l) = -\log_{10}\left(\frac{|\mathrm{PsyMis}(l)| + 1}{|\mathrm{Concepts}(l)|}\right) \tag{4}$$

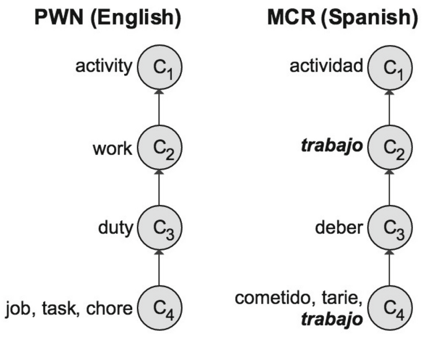


Fig. 5. A psycholinguistic mistake in Spanish.

$$\mathrm{LanQua}(l) = \frac{\mathrm{AbsLanQua}(l)}{\mathrm{AbsLanQua}(\mathrm{English})} \tag{5}$$

$$\mathrm{AvgDis}(l) = \frac{\sum_{x \in \mathrm{PsyMis}(l)} \mathrm{dis}(x)}{|\mathrm{PsyMis}(l)|} \tag{6}$$

where PsyMis(l) is the set of psycholinguistic mistakes in l, AbsLanQua(l) and LanQua(l) are the *Absolute Language quality* and the *Language quality* of l, respectively. The number of mistakes varies a lot, going from the fourteen mistakes of the PWN English to the thousands of mistakes of other languages. The Log-based definition of AbsLanQua is meant to alleviate this problem (see Tables 2and 3). English is taken to be the reference to which we normalize the quality of the other languages. *dis*(x) is the number of intermediate nodes between two concepts generating the psycholinguistic mistake x, for instance, in Fig. 5, *dis*(trabajo) = 2. The *Average Distance* AvgDis measures the average distance for a language. As from Table 3, this distance is around 1 for most languages with the exception of English where it is 3.42, which provides even more evidence of the large gap in quality between the PWN English and any other language.

Figures 6 and 7 compare the incompleteness and quality values of the languages in the UKC, where the ten languages in Table 3 are explicitly marked with their ISO names, as from Table 3.

Figure 6 shows that most languages have a low quality, below 0.4, and that the most developed languages (the ones with LanInc below 0.7), with the exception of English, have even lower values. In other words, the mistakes grow with the size of the language itself. Figure 7 compares incompleteness and the absolute number of mistakes. Here, the majority of languages is below the dashed line making even more explicit how the number of mistakes grows with the size of the resource.

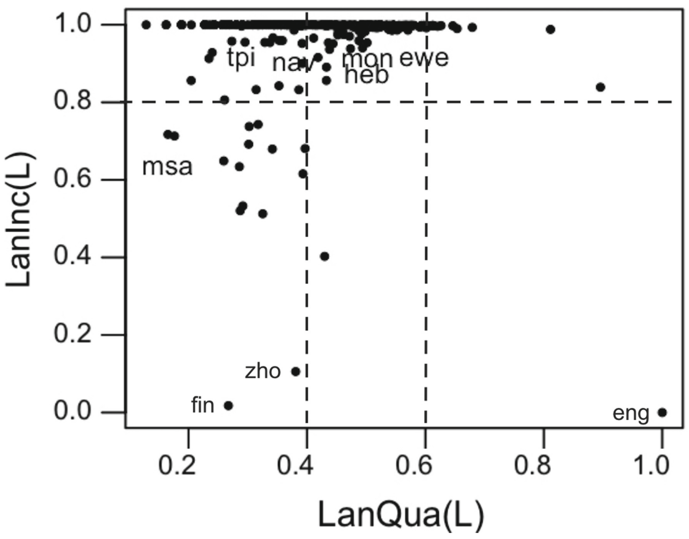


Fig. 6. Language incompleteness vs language quality.

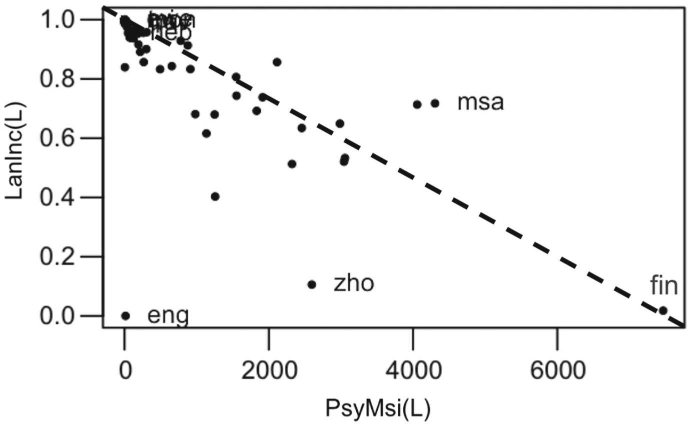


Fig. 7. Language incompleteness vs psycholinguistic mistakes.

7 Related Work

As far as we know, the stratification of meaning into words, synsets and concepts and the resulting three-layer architecture of the UKC has never been proposed before. Relevant work has been done, however, in the development of large scale multilingual resources.

BabelNet [27] is the largest multilingual lexico-semantic resource, obtained from an automatic integration of several resources. Currently, this resource covers 271 languages, 6 million concepts and millions of words. The design decisions underlying BabelNet and the UKC are fundamentally different. The most important difference is that in the UKC we do not allow the addition of entities with the exception of those (a very small minority) which are mentioned in glosses. The main motivation for this decision is that we want to keep the UKC inherently linguistic and focused on concepts. Notice how the number of entities is essentially unbound, order or magnitude bigger than that of concepts and, modulo a (still large) number of exceptions (e.g., the names of famous locations or people), inherently bound to the local cultures. The second main difference is our focus on quantifiable high quality, a property which is hard to maintain when performing automatic resource integration.

Lately, a lot of work has focused on the creation of a *Global Wordnet Grid*, which is currently being instantiated in the *Open Multilingual Wordnet* [28], and whose goal is to link the concepts from different Wordnets. Towards this goal, the Collaborative InterLingua Index (CILI) [29] aims at enabling the coordination among multiple loosely coordinated Wordnets. Finally, as part of the development of Global Wordnet Grid, the work described in [30] has recently introduced the idea of a central registry of concepts. This latter idea is somewhat related to our idea of clustering the synsets with the same meaning under the same concept. However, differently from us, in this work, the different languages are only loosely coupled and there seems no easy way to use the different languages inside a single application. Furthermore, this work seems still somewhat early stage, in terms of number of languages which has been integrated so far, and also, in terms of quality control of the resource.

The *diversity among languages* has been extensively studied in the fields of *Historical* and *Comparative Linguistics* [31] with, however, major limitations due to the problem that the data sets are very small (in the order of tens of elements). The work described in [5] provides a language diversity aware algorithm which can distinguish between homographs and polysemes. It is a first example of how the UKC paves the way to large scale quantitative studies of language diversity also, but not only, towards the production of high quality language resources.

8 The Way Ahead

The UKC is at a mature stage of development, but with still a lot of work to be done. We foresee the following areas of further research: (i) the development of a computational diversity-aware theory of meaning based on the notion of concept defined above, (ii) the use of the UKC for the development of quantitative studies of language diversity, starting from an in depth analysis of lexical gaps, (iii) the further development of the UKC, as a community effort, both in terms of new languages and of enrichment of the existing languages, (iv) a refinement of the CC aimed to aligning lexical concepts with the results of perception and, last but not least, (v) the extensive use of the UKC in language aware applications, with a focus on large scale video classification.

Acknowledgments. The first version of the UKC was developed by Ilya Zaihrayeu, around 2004. This implementation was revised many times, most often as a joint effort between Ilya and Marco Marasca. Since the beginning, the UKC has been designed with the goal of supporting the automation of reasoning based on information extracted from text, the original goal being the matching of ontologies [32]. We thank the many postdocs and PhD students who have extensively used the UKC in their research.

The current work is supported by QROWD (http://qrowd-project.eu), a Horizon 2020 project, under Grant Agreement No. 732194. The second author is supported by the ESSENCE Marie Curie Initial Training Network, funded by the European Commission's 7th Framework Programme under grant agreement no. 607062.

References

1. Miller, G.A., Beckwith, R., Fellbaum, C., Gross, D., Miller, K.J.: Introduction to wordnet: an on-line lexical database. Int. J. Lexicogr. **3**(4), 235–244 (1990)
2. Vossen, P.: Introduction to EuroWordNet. Comput. Humanit. **32**(2–3), 73–89 (1998)
3. Pianta, E., Bentivogli, L., Girardi, C.: Multi-wordnet: developing an aligned multilingual database. In: Proceedings of the First International Conference on Global WordNet, Mysore, India, pp. 21–25, January 2002
4. Gonzalez-Agirre, A., Laparra, E., Rigau, G.: Multilingual central repository version 3.0. In: LREC, pp. 2525–2529 (2012)
5. Giunchiglia, F., Batsuren, K., Bella, G.: Understanding and exploiting language diversity. In: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI-17), pp. 4009–4017 (2017)
6. Henrich, J., Heine, S.J., Norenzayan, A.: The weirdest people in the world? Behav. Brain Sci. **33**(2–3), 61–83 (2010)
7. Bella, G., Giunchiglia, F., McNeill, F.: Language and domain aware lightweight ontology matching. In: Web Semantics: Science, Services and Agents on the World Wide Web (2017)
8. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: a large-scale hierarchical image database. In: CVPR 2009 (2009)
9. Deng, J., Russakovsky, O., Krause, J., Bernstein, M., Berg, A.C., Fei-Fei, L.: Scalable multi-label annotation. In: ACM Conference on Human Factors in Computing Systems (CHI) (2014)
10. Giunchiglia, F., Fumagalli, M.: Concepts as (recognition) abilities. In: Formal Ontology in Information Systems: Proceedings of the 9th International Conference (FOIS 2016), pp. 153–166 (2016)
11. Giunchiglia, F., Fumagalli, M.: Teleologies: objects, actions and functions. In: Proceedings of the 36th International Conference on Conceptual Modeling (ER 2017) (2017)
12. Millikan, R.G.: On Clear and Confused Ideas: An Essay About Substance Concepts. Cambridge University Press, Cambridge (2000)
13. Kjellmer, G.: Lexical gaps. Lang. Comput. **48**(1), 149–158 (2003)
14. Bentivogli, L., Pianta, E.: Looking for lexical gaps. In: Proceedings of the Ninth EURALEX International Congress, pp. 8–12. Universität Stuttgart, Stuttgart (2000)
15. Cvilikaitė, J.: Lexical gaps: resolution by functionally complete units of translation. Darbai ir dienos **2006**(45), 127–142 (2006)
16. Lehrer, A.: Notes on lexical gaps. J. Linguist. **6**(2), 257–261 (1970)

17. Crowley, T., Bowern, C.: An Introduction to Historical Linguistics, 4 edn. Oxford University Press, Oxford (2010)
18. Croft, W.: Typology and Universals. Cambridge University Press, Cambridge (2002)
19. Rijkhoff, J., Bakker, D., Hengeveld, K., Kahrel, P.: A method of language sampling. Studies in language. Int. J. sponsored Found. "Found. Lang. **17**(1), 169–203 (1993)
20. Bell, A.: Language samples. Univers. Hum. Lang. **1**, 123–156 (1978)
21. McMahon, A., McMahon, R.: Language Classification by Numbers. Oxford University Press on Demand, Oxford (2005)
22. Swadesh, M.: Towards greater accuracy in lexicostatistic dating. Int. J. Am. Linguist. **21**(2), 121–137 (1955)
23. Swadesh, M.: The Origin and Diversification of Language. Transaction Publishers, Piscataway (1971)
24. Greenberg, J.H.: Univers. Lang. (1966)
25. Youn, H., et al.: On the universal structure of human lexical semantics. Proc. Natl. Acad. Sci. **113**(7), 1766–1771 (2016)
26. Miller, G.A.: Nouns in wordnet: a lexical inheritance system. Int. J. Lexicogr. **3**(4), 245–264 (1990)
27. Navigli, R., Ponzetto, S.P.: Babelnet: building a very large multilingual semantic network. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pp. 216–225. Association for Computational Linguistics (2010)
28. Bond, F., Foster, R.: Linking and extending an open multilingual wordnet. In: ACL, vol. 1, pp. 1352–1362 (2013)
29. Bond, F., Vossen, P., McCrae, J.P., Fellbaum, C.: Cili: the collaborative interlingual index. In: Proceedings of the Global WordNet Conference, vol. 2016 (2016)
30. Vossen, P., Bond, F., McCrae, J.: Toward a truly multilingual globalwordnet grid. In: Proceedings of the Eighth Global WordNet Conference, pp. 25–29 (2016)
31. Von Fintel, K., Matthewson, L.: Universals in semantics. Linguist. Rev. **25**(1–2), 139–201 (2008)
32. Giunchiglia, F., Autayeu, A., Pane, J.: S-match: an open source framework for matching lightweight ontologies. Semant. Web **3**(3), 307–317 (2012)

Analysis of Speeches in Indian Parliamentary Debates

Sakala Venkata Krishna Rohit(✉) and Navjyoti Singh

Center for Exact Humanities (CEH),
International Institute of Information Technology, Hyderabad, India
rohitsakala@gmail.com

Abstract. With the increasing usage of internet, more and more data is being digitized including parliamentary debates but they are in an unstructured format. There is a need to convert them into structured format for linguistic analysis. Much work has been done on parliamentary data such as Hansard, American congressional floor-debate data on various aspects but less on pragmatics. In this paper, we provide a dataset for synopsis of Indian parliamentary debates and perform stance classification of speeches i.e identifying if the speaker is supporting the bill/issue or is against it. We also analyze the intention of the speeches beyond mere sentences i.e pragmatics in the parliament. Based on thorough manual analysis of the debates, we developed an annotation scheme of 4 mutually exclusive categories to analyze the purpose of the speeches: to find out ISSUES , to BLAME, to APPRECIATE and for CALL FOR ACTION. We have annotated the dataset provided, with these 4 categories and conducted preliminary experiments for automatic detection of the categories. Our automated classification approach gave us promising results.

Keywords: India · Parliamentary debates · Stance classification · Debate analysis · Dataset

1 Introduction

As the world moves towards increasing forms of digitization, the creation of text corpora has become an important activity for NLP and other fields of research. Parliamentary data is a rich corpus of discourse on a wide array of topics. The Lok Sabha[1] website[2] provides access to all kinds of reports, debates, bills related to the proceedings of the house. Similarly, the Rajya Sabha[3] website also contains debates, bills, reports introduced in the house. The Lok Sabha website also contains information about members of the parliament who are elected by the people and debate in the house. Since the data is unstructured, it cannot be

[1] Lok Sabha is the lower house of the Indian Parliament.
[2] http://164.100.47.194/Loksabha/Debates/debatelok.aspx.
[3] Rajya Sabha is the upper house of the Indian Parliament.

computationally analyzed. There is a need to shape the data into a structured format for analysis. This data is important as it can be used to visualize person, party and agenda level semantics in the house.

The data that we get from parliamentary proceedings has presence of sarcasm, interjections and allegations which makes it difficult to apply standard NLP techniques [1]. Members of the parliament discuss various important aspects and there is a strong purpose behind every speech. We wanted to analyze this particular aspect. Traditional polar stances (for or against) do not justify for the diplomatic intricacies in the speeches. We created this taxonomy to better understand the semantics i.e the pragmatics of the speeches and to give enriched insights into member's responses in a speech. The study of the speaker's meaning, not focusing on the phonetic or grammatical form of an utterance, but instead on what the speaker's intentions and beliefs are is pragmatics. Pragmatics is a sub-field of linguistics and semiotics that studies the ways in which context contributes to meaning.

After thorough investigation of many speeches we found that the statements made by members cannot be deemed strictly "for or against" a bill or government. A person maybe appreciating a bill or government's effort in one part of a speech but also asking attention to other contentious issues. Similarly, a person criticizing government for an irresponsible action could be giving some constructive suggestions elsewhere. A political discourse may not always be polar and might have a higher spectrum of meaning. After investigating and highlighting statements with different intentions we came up with a minimal set of 4 mutually exclusive categories with different degrees of correlation with the traditional two polar categories (for and against). It is observed that any statement by a participating member will fall into one of these categories namely - Appreciation, Call for Action, Issue, Blaming (Table 1).

Table 1. Degrees of co-relation

Stance/category	Appreciation	CallForAction	Issue	Blame
For	Medium	High	Low	Low
Against	Low	High	High	Low

For example, if the debate consists of more of issues, one can infer that the bill is not serving the its purpose in a well manner. Also, this preliminary step will lead to new areas of research such as detection of appreciation, blame in similar lines of argument mining which is evolving in the recent years in the field of linguistics. We will quote portions of a few speeches which will give an idea of the data being presented:

"*This city has lost its place due to negligence of previous governments and almost all industries have migrated from here and lack of infrastructure* *facilities, business is also losing its grip. It is very unfortunate that previous UP Governments also did not do any justice to this city.*"

- Shri Devendra Singh Bhole, May 03, 2016

As evident, the speaker is clearly blaming the previous governments for negligence on the city. In this sense the data is very rich and a lot of linguistic research is possible. Researchers can work on different aspects such as detection of critique made by members, suggestions raised by members etc. Given the data, it can be used for rhetoric, linguistic, historical, political and sociological research. Parliamentary data is a major source of socially relevant content. A new series of workshops are being conducted for the sole purpose of encouraging research in parliamentary debates ParlClarin.

As a preliminary step, we created four major categories of the speeches spoken by the parliament members. The definitions and examples of the four categories are explained in the below tables respectively. The examples are taken from a debate on NABARD bill in Lok Sabha (Table 2).

Table 2. Definition of the categories

Category	Definition
Issue	Raise problems in general which need attention
Blame	Blaming the government or the opposition government or policies
Appreciate	Appreciating and justifying policies , governments by mentioning the benefits and efforts of it
Call for action	Speeches in which members suggest, request for new laws/proposals

A speech can be labelled with multiple categories as members can appreciate and raise issues in the same speech. The following points are the contributions of this paper:

– A dataset of Indian Parliamentary Debates.
– Creation of categories for classifying the purpose of the speech acts i.e pragmatics
– Preliminary experiment on automatic detection of the categories.
– Stance classification of speeches.

2 Related Work

Many linguists around the globe are concentrating on creation of parliamentary datasets. [2] gives an overview of the parliamentary records and corpora from countries with a focus on their availability through Clarin infrastructure. A dataset of Japanese Local Assembly minutes was created and analyzed for

Table 3. Example statements of the categories

Category	Example statements
Issue	....The present scenario is that credit from nationalized banks and from financial institutions is not available. That is why there is a crisis in the agrarian economy.....
Blame	.....The policy of the Government is going in one direction and the banks which come under the Finance Ministry are going in a totally different directions.....
Appreciate	.....The Government's vision of 'Sabka Sath Sabka Vikas' comes to the fore through this Bill......This will usher in not only the rural development but also leave the rural folk to their self-reliance. At the same time, the rural development will prevent them from migrating to the cities and everybody will get employment.....
Call for action	.....So, NABARD should come forward with new credit and interest policy so that more farmers could avail of the loan.....That is why, the coverage of NABARD and its access to the real masses or the poor people should be further expanded.

statistical data such as number of speakers, characters and words [3]. [4] created a highly multilingual parallel corpus of European parliament and demonstrated that it is useful for statistical machine translation. Parliamentary debates are full of arguments. Ruling party members refute the claims made by opposition party members and vice versa. Members provide strong arguments for supporting their claim or refuting other's claim. Analyzing argumentation from a computational linguistics point of view has led very recently to a new field called argumentation mining [5]. One can perform argument mining on these debates and analyze the results. [6] worked on detecting perspectives in UK political debates using a Bayesian modelling approach. [7] worked on claim detection from UK political debates using both linguistic features text and features from speech.

Stance classification is a relatively new and challenging approach to deepen opinion mining by classifying a user's stance in a debate i.e whether he is for or against the topic. [8]. [9] addressed the question of whether opinion mining techniques can be used on Congressional debates or not. [12] worked on stance classification of posts in online debate forums using both structural and linguistic features. [10] trained a svm [11] classifier with features of unigrams, bigrams and trigrams to predict whether a sentence is in agreement or disagreement and achieved an F-score of 0.55 for agreement and 0.81 for disagreement on the evaluation set. No one has worked on classifying speeches based on their purpose. This is the first novel work towards this aspect.

3 DataSet

Our dataset consists of synopsis of debates in the lower house of the Indian Parliament (Lok Sabha). The dataset consists of:

– 19 MB approx.
– 189 sessions
– 768 debates
– 5575 speeches
– 1586838 words
– Sessions from 2014 to 2017.

In Lok Sabha, a session is referred to as all the debates held in a particular cycle of sitting. There are 55 debate types[4] identified by the Lok Sabha. Table 3 identifies some of the debate types we have considered and their frequency between the years 2014 and 2017. We opted out debate types which do not occur regularly. Each debate type has its own style of proceedings. For example, in the debate type "Government Bills", a minister places a bill on the table and discussion is carried out on the bill where as in the debate type "Matter under 377", each speaker raises an issue of which he is concerned of but no discussion is done on the issues.

Table 4. Different debate types in Lok Sabha.

Debate type	Occurrence count
Matter Under 377	124
Submission members	101
Government bills	87
Discussion	42
General	41
References	37
Minister statement	33
Statutory resolutions	27
Private member Bills	15
President thanks	12
Motion	6

3.1 Creation

The creation of the dataset involved 3 steps. The first step was to scrap the pdf files from the Lok Sabha website. Each pdf file is a session. The second step was to convert the pdf files into text files for easy parsing. The challenge here was to convert this unstructured information into a structured format. The third step is to extract relevant data using pattern matching. We developed a software parser[5] for extracting the entities such as date, debate type, member name and

[4] http://164.100.47.194/Loksabha/Debates/DebateAdvSearch16.aspx
[5] https://github.com/rohitsakala/synopsisDebateParser

speech. We used regex, pattern matching code to find out patterns from the text file. For example to segregate a speaker's name from his speech, we used:

```
re.split(":")
```

as name of the speaker and his/her speech is separated by a colon. An example pdf can be accessed using this URL. Right now, member name and bill name are needed to be stored manually which we plan to automate too. Sometimes the pattern matching fails due to irregularities in the pdf as those were written by humans though they were negligible. We stored the structured data into a Mongo database as different debate types have different schema. The database consists of the following tables:

- `Sessions`: all the debates happened on a particular day with date, secretary general name.
- `Members`: information about the members/speakers of the parliament i.e name and party affiliation.
- `Debates`: contains the member id and the corresponding speeches, summaries and keywords.
- `Bills`: the name of the bill.
- `Debate Type`: the name of the debate type.

The software parser developed is very generic. As new sessions are being added on the Lok Sabha website, the software parser automatically identifies them, parses it and stores the structured data in the database. The database has been hosted in a online database hosting site, mLab. The mongo shell can be accessed using this command in any linux machine which has mongo installed.

```
mongo ds235388.mlab.com:35388/synopsis -u public -p public
```

3.2 Annotation

We have annotated 1201 speeches with the four categories mentioned above, on the speeches. We also annotated stances of the speakers towards the bill/issue that is being debated on. There are two stances one is *for* and other is *against*. The statistics of the annotated data is shown in Table 4.

Table 5. Statistics of annotated data

Categories	Count
Issue	589
Blame	147
Appreciate	522
Call for Action	930
For	919
Against	282

Two humanities students were involved in the annotation of the four categories on 1201 speeches. The annotator agreement is shown in Table 5 and is evaluated using two metrics, one is the Kohen's Kappa [20] and other is the inter annotator agreement which is the percentage of overlapping choices between the annotators.

Table 6. Inter annotator agreement metrics of annotated data

Category	Kohen's Kappa	Inter annotator agreement
Issue	0.67	0.84
Blame	0.65	0.90
Appreciate	0.88	0.94
Call for Action	0.46	0.92

The inter annotator agreement for the stance categories were 0.92. The high values of inter annotator scores clearly explain how easy it was to delineate each category. It also signifies that the definition of the category that needed to be annotated, were very clear.

3.3 Keywords and Summarization

We have used TextRank which is an extractive summariser [18] for summarizing the entire debate and for finding keywords in the debate. TextRank is a graph based ranking model for text processing specifically KeyPhrase Extraction and Sentence Extraction. TextRank performs better in text summarization using graph based techniques [19]. We added these two extra fields i.e the keywords extracted by TextRank and the summary created by TextRank in the debates collection. An example summary is:

The last National Health Policy was framed in 2002. The Policy informs and prioritizes the role of the Government in shaping health systems in all its dimensions investment in health, organization and financing of health care services, prevention of diseases and promotion of good health through cross-sectoral *action, access to technologies, developing human resources, encouraging medical pluralism, building the knowledge base required for better health, financial protection strategies and regulation and progressive assurance for health. The Policy aims for attainment of the highest possible level of health and well-being for all at all ages, through a preventive and promotive health care orientation in all developmental policies, and universal access to good quality health care services without anyone having to face financial hardship as a consequence. The Policy seeks to move away from Sick-Care to Wellness, with thrust on prevention and health care promotion. Before this, the Policy was for the Sick-Care Health Policy. Now we are making it Promotional and Preventive Health Policy. While the policy seeks to reorient and strengthen the public health systems, it also looks afresh at strategic purchasing from the private sector and leveraging their strengths to achieve national health goals. As a crucial component, the policy proposes raising public health expenditure to 2.5 per cent of the GDP in a time bound manner. The Policy has also assigned specific quantitative targets aimed at reduction of disease prevalence/incidence under three broad components viz., (a) health status and programme impact, (b) health system performance, and (c) health systems strengthening, aligned to the policy objectives. To improve and strengthen the regulatory environment, the policy seeks putting in place systems for setting standards and ensuring quality of health care. The policy advocates development of cadre of mid-level service providers, nurse practitioners, public health cadre to improve availability of appropriate health human resource. The policy also seeks to address health security and Make in India for drugs and devices. It also seeks to align other policies for medical devices and equipment with public health goals.*

3.4 Detection of Polarity

To detect the polarity of each speech, we have used VADER [17] sentiment analysis tool. The tool uses a simple rule-based model for general sentiment analysis and generalizes more favorably across contexts than any of many benchmarks such as LIWC and SentiWordNet. The tool takes as input a sentence and gives a score between -1 and 1. The polarity of a speech is calculated by taking the sum of the polarities of the sentences. If the sum is greater than zero, then it is classified as *positive*, if it is less than zero, then it is classified as *negative* and if it is equal to zero then it is classified as *neutral.* The statistics of the data is presented in Table 6.

Table 7. Sentiment polarity of speeches

Category	Count
Positive	4006
Negative	1457
Neutral	112
Total	5575

3.5 Examples

– A Document in session collection[6].

```
{
"_id" : ObjectId("5a4255c789..") ,
"indianDate" : "Vaisakha 9,1938(Saka)",
"debates" : {
"5999649837.." : ObjectId("5a425b5..") ,
"5999644a37.." : ObjectId("5a425b06..")
}
"englishDate" : "Friday,April 29,2016",
"houseName" : "LOK SABHA",
"secretaryGeneralName" : "ANOOP MISHRA"
}
```

The _id is the unique key assigned by the mongo database. The keys[7] in the debates key represent the debate types from the debate types collection. The values of the debates key refer to the corresponding debates in the debates collection.

– A Document in member collection. The table consists of name of the member spoken, the house of the parliament and the party to which he is affiliated.

```
{
"_id" : ObjectId("59a8e0e983"),
"name" : "Dharambir Singh,Shri",
"house" : "Lok Sabha",
"party" : "BJP"
}
```

– A Document in bill collection. The table consists of the bill name.

```
{
"_id" : ObjectId("59de525596...") ,
"name" : "THE COMPENATION BILL, 2016"
}
```

– A Document in debates collection of debate type Submission Members. The table consists of all the speeches made in a particular debate in an order with summary and keywords from TextRank.

[6] A table in mongo database is called collection.
[7] A key refers to the key in the json data type.

```
{
"_id" : ObjectId("5a42539889..") ,
"topic" : "Flood situation in ..." ,
"keywords" : "water state ... ",
"summary" : "..." ,
"speeches" : {
  "1" : {
  "speech" : "In Tamil Nadu and in..." ,
  "memberId" : "59a92d88a0b4..." ,
  "polarity" : "Negative"
  },
  "2" : {
  "speech" : "We all have witness..." ,
  "memberId" : "59cbc3ef6636..." ,
  "polarity" : "Positive"
  },
  "3" : {
          ...
  }
  ...
  ...
}
```

The memberId refers to the _id in the member's collection.

4 Experiment

In this section, we deal with two tasks, task one is the classification of the stances the speakers take and task two is the classification of categories based on purpose. Stance classification differs from sentiment analysis. For instance, the number of speeches that were annotated as *for* i.e 919 had only 719 labelled as *positive* and the number of speeches that were annotated as *against* i.e 282 had only 89 as *negatively* labelled. So, these statistics clearly indicate the difference between polarity detection and stance classification.

Text classification is a core task to many applications, like spam detection, sentiment analysis or smart replies. We used fastText and SVM [16] for preliminary experiments. We have pre-processed the text removing punctuation's and lowering the case. Facebook developers have developed fastText [13] which is a library for efficient learning of word representations and sentence classification. The reason we have used fastText is because of its promising results in [14].

We divided our training and testing data in the ratio of 8:2 for classification. As mentioned above we used fastText and SVM for both the classification tasks. We report accuracy for each class as it is a multi-label classification problem. The results are shown in Table 7 and Table 8. Also, the parameters used for fastText is described in Table 9.

Table 8. Accuracy score for classification task 1

Task/Metric	fastText	SVM
For/Against	**0.80**	0.76

We have not used hs (Hierarchical Soft-max) for binary classification, instead used regular softmax as it was giving better results in fastText.

Table 9. Accuracy score for classification task 2

Task 2/Metric	fastText	SVM
Call for Action	**0.745**	0.72
Issue	**0.604**	0.56
Blame	0.783	**0.84**
Appreciate	**0.679**	0.62

Table 10. Parameters used for fastText algorithm

Parameter	Value
Learning Rate	0.8
Word Dimension	100
n-gram	2
Epoch	100
Loss Function	hs

For SVM, the features were the word vectors trained using word2vec [15] with dimension size of 300 whereas for fastText, the features were the word vectors trained using character n-gram embedding. We have achieved considerably good results. We plan to annotate more and check if the accuracy increase any further. The limitation that we feel is the number of annotations being done. We approached the classification problem as one vs rest classification problem. We performed the classification on document level. Later we would like to analyze at sentence level. The least accuracy was for Issue category and the highest is for Blame category. This research will inspire researchers to take on further research on mining appreciation, blaming from text in lines with the ongoing approaches of argument mining, hate speech, sarcasm generation etc.

As we increase the number of epochs in the fastText, the scores also increase as evident from Table 10, but the increase stops after 25 epochs (Table 11).

Table 11. Analysis of fastText for Task 2 with varying epochs for Call for Appreciate category

Epochs	Accuracy
5	0.579
10	0.65
25	0.6916
50	0.6708
100	0.679

5 Conclusion

In this paper, we presented a dataset of synopsis of Indian parliamentary debates. We developed a generic software parser for the conversion of unstructured pdfs into structured format i.e into a relational database using mongo database software. We analyzed the purpose of the speeches of the member of parliament and categorized them into 4 major categories and provided statistics of the categories. We also tried to identify them automatically using fastText algorithm and provided the results. The analysis is done for understanding the purpose of the speeches in the parliament. We also presented our results on binary stance classification of the speeches whether the member is in favour of the debate topic or not.

6 Future Work

In future, we would like to increase the size of the dataset by including sessions of previous years which are not yet digitized. Sessions before 2009 are yet to be digitalised by the Lok Sabha editorial of India. Also we plan to include Rajya Sabha debates into the dataset. We have used fastText for classifying categories. We plan to develop a set of features to increase the accuracy of the classification task as we believe that features like party affiliation will have greater impact and experiment with other machine learning approaches.

TextRank is used for summarization. We feel that for political debates, summarization should emphasize on arguments made by members unlike TextRank. In the whole debate, a lot of themes are raised by the members. The debate revolves around these themes. So, developing a model for thematic summarization with arguments will capture the complete picture of the entire debate unlike TextRank. We plan to do this as our future work on these debates. A short summary of the important themes discussed with its arguments will benefit journalists, newspaper editors, common people etc.

References

1. Onyimadu, O., Nakata, K., Wilson, T., Macken, D., Liu, K.: Towards sentiment analysis on parliamentary debates in Hansard. In: Kim, W., Ding, Y., Kim, H.-G. (eds.) JIST 2013. LNCS, vol. 8388, pp. 48–50. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-06826-8_4
2. Fiser, D., Lenardic, J.: Parliamentary Corpora in the CLARIN infrastructure (2017). https://www.clarin.eu/sites/default/files/thematic-session-2-lenardic.pdf
3. Yasutomo, K., et al.: Creating Japanese political corpus from local assembly minutes of 47 prefectures. In: Proceedings of the 12th Workshop on Asian Language Resources (ALR12), pp. 78–85 (2016)
4. Hajlaoui, N., Kolovratnik, D., Vayrynen, J., Steinberger, R., Varga, D.: DCEP - digital corpus of the European Parliament. In: Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC 2014), Reykjavik, Iceland (2014)
5. Green, N.S.: Towards creation of a corpus for argumentation mining the biomedical genetics research literature. In: ArgMining@ACL (2014)
6. He, Y., Vilares, D.: Detecting perspectives in political debates. In: EMNLP (2017)
7. Lippi, M., Torroni, P.: Argument mining from speech: detecting claims in political debates. In: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI 2016), pp. 2979–2985. AAAI Press (2016)
8. Addawood, A., Schneider, J., Bashir, M.: Stance classification of twitter debates: the encryption debate as a use case. In: Proceedings of the 8th International Conference on Social Media and Society, Article 2, 10 p. ACM, New York (2017). https://doi.org/10.1145/3097286.3097288
9. Thomas, M., Pang, B., Lee, L.: Get out the vote: determining support or opposition from congressional floor-debate transcripts. In: Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP 2006), pp. 327–335. Association for Computational Linguistics, Stroudsburg (2006)
10. Kerren, A., Paradis, C., Skeppstedt, M., Sahlgren, M.: Unshared task: (dis)agreement in online debates. In: ArgMining@ACL (2016)
11. Pedregosa, F., et al.: Scikit-learn: machine Learning in Python. J. Mach. Learn. Res. 2825–2830 (2011)
12. Getoor, L., Sridhar, W.M.D.: Collective Stance Classification of Posts in Online Debate Forums (2014)
13. Armand, J., Edouard, G., Piotr, B., Tomas, M.: Bag of Tricks for Efficient Text Classification (2016)
14. Akshita, J., Radhika, M.: When does a compliment become sexist? Analysis and classification of ambivalent sexism using twitter data. In: Proceedings of the Second Workshop on NLP and Computational Social Science, pp. 7–16. Association for Computational Linguistics (2017)
15. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Proceedings of the 26th International Conference on Neural Information Processing Systems (NIPS 2013), vol. 2, pp. 3111–3119. Curran Associates Inc., USA (2013)
16. Hearst, M.A.: Support vector machines. IEEE Intell. Syst. **13**(4), 18–28 (1998). https://doi.org/10.1109/5254.708428
17. Gilbert, E., Hutto, C.J.: VADER: a parsimonious rule-based model for sentiment analysis of social media text. In: ICWSM (2014)
18. Mihalcea, R., Tarau, P.: TextRank: Bringing Order Into Texts (2004)

19. Mihalcea, R.: Graph-based ranking algorithms for sentence extraction, applied to text summarization. In: Proceedings of the ACL 2004 on Interactive Poster and Demonstration Sessions (ACLdemo 2004), Article 20. Association for Computational Linguistics, Stroudsburg (2004). https://doi.org/10.3115/1219044.1219064
20. Cohen, J.: A coefficient of agreement for nominal scales. Educ. Psychol. Measur. **V20**, 37–46 (1960)

Enrichment of OntoSenseNet: Adding a Sense-annotated Telugu Lexicon

Sreekavitha Parupalli(✉) and Navjyoti Singh

Center for Exact Humanities (CEH) International Institute of Information Technology, Hyderabad, India
sreekavitha.parupalli@research.iiit.ac.in, navjyoti@iiit.ac.in

Abstract. The paper describes the enrichment of OntoSenseNet- a verb-centric lexical resource for Indian Languages. This resource contains a newly developed Telugu-Telugu dictionary. It is important because native speakers can better annotate the senses when both the word and its meaning are in Telugu. Hence efforts are made to develop a soft copy of Telugu dictionary. Our resource also has manually annotated gold standard corpus consisting 8483 verbs, 253 adverbs and 1673 adjectives. Annotations are done by native speakers according to defined annotation guidelines. In this paper, we provide an overview of the annotation procedure and present the validation of our resource through inter-annotator agreement. Concepts of sense-class and sense-type are discussed. Additionally, we discuss the potential of lexical sense-annotated corpora in improving word sense disambiguation (WSD) tasks. Telugu WordNet is crowd-sourced for annotation of individual words in synsets and is compared with the developed sense-annotated lexicon (OntoSenseNet) to examine the improvement. Also, we present a special categorization (spatio-temporal classification) of adjectives.

Keywords: OntoSenseNet · Sense-annotated corpora · Word sense disambiguation

1 Introduction

Lexically rich resources form the foundation of all natural language processing (NLP) tasks. Maintaining the quality of resources is thus a high priority issue [5]. Hence, it is important to enhance and maintain the lexical resources of any language. This is of significantly more importance in case of resource poor languages like Telugu [19].

WordNet is a vast repository of lexical data and it is widely used for automated sense-disambiguation, term expansion in IR systems, and the construction of structured representations of document content [12]. First WordNet among the Indian languages was developed for Hindi. WordNets for 16 other Indian languages are built from Hindi WordNet applying expansion approach [1].

WSD can be characterized as a task that emphasizes on evaluating the right sense of a word in its particular context. It is a critical pre-processing step in

https://doi.org/10.1007/978-3-031-23793-5_21

data extraction, machine translation, question answering systems and numerous other NLP tasks. Vagueness in word sense emerges when a specific word has multiple conceivable senses. Finding the right sense requires exhaustive information of words. This additional information be call as *intentional* meaning of the word. Meaning can be discussed as sense (intensional meaning) and reference (extensional meaning) [6]. The meaning of a word, from ontological viewpoint, can be understood based on its participation in classes, events and relations. We use a formal ontology that is developed to computationally manipulate language at the level of meanings which have an intrinsic form [13].

The paper is organized as follows. Section 2 discusses available lexical resources for Telugu and several types of WSD tasks that were previously developed. Section 3 describes our dataset and shows the statistics of available lexical resources.Section 4 talks about the ontological classification that was formalized for the annotation purpose by [13]. Section 5 describes annotation guidelines and explains the procedure of manual annotation by expert native speakers. Section 9 concludes the paper and Sect. 10 presents the scope of future work in the domain.

IAST based transliteration[1] for Telugu script has been employed in the paper.

2 Related Work

Before understanding the tasks that are performed, it is important to understand and analyze the available resource thoroughly. Hence this section discusses the previous work that was done in this domain.

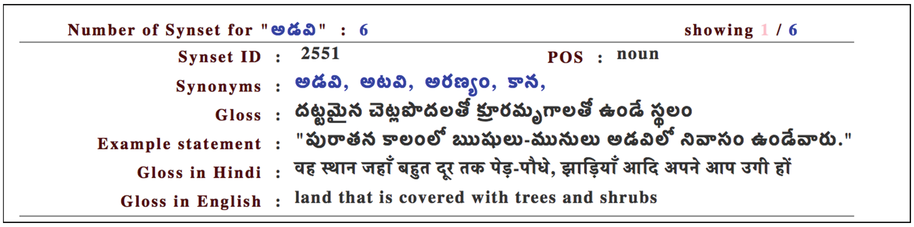
2.1 Telugu WordNet

Telugu WordNet is developed as a part of IndoWordNet[2] at CFILT [2], which is considered as the most exhaustive set of multilingual lexical assets for Indian languages. It consists of 21091 synsets in total. This total includes 2795 verb synsets, 442 adverb synsets, 5776 adjective synsets. Telugu WordNet captures several other semantic relations such as hypernymy, hyponymy, holonymy, meronymy, antonymy. For every word in the dictionary it provides synset ID, parts-of-speech (POS) tag, synonyms, gloss, example statement, gloss in Hindi, gloss in English. An example of an entry in the IndoWordNet database is shown in Fig. 1.

2.2 Ontological Issues in WordNet

WordNet is a language specific resource and it varies from language to language. However, any WordNet can be considered an ontology through the hypernymy-hyponymy relations that are present in it. WordNet of any language leaves a few loop holes that other ontologies can fill [1]. By summarizing the following [11], [7], [15], we state the four major problems:

[1] http://www.learnsanskrit.org/tools/sanscript.
[2] http://www.cfilt.iitb.ac.in/indowordnet/index.jsp.



Number of Synset for "అడవి" : 6 showing 1 / 6

Synset ID : 2551	POS : noun
Synonyms :	అడవి, అటవి, అరణ్యం, కాన,
Gloss :	దట్టమైన చెట్లపొదలతో క్రూరమృగాలతో ఉండే స్థలం
Example statement :	"పురాతన కాలంలో ఋషులు-మునులు అడవిలో నివాసం ఉండేవారు."
Gloss in Hindi :	वह स्थान जहाँ बहुत दूर तक पेड़-पौधे, झाड़ियाँ आदि अपने आप उगी हों
Gloss in English :	land that is covered with trees and shrubs

Fig. 1. Example entry in the IndoWordNet database

- *Confusing concepts with individuals:* WordNet synsets do not distinguish between universal and the particular instance of a concept. For example, both 'aḍavi(forest)' and 'ceṭṭu(tree)' are considered as concept.
- *Lexical gap:* A language may not have an indigenous lexeme to describe a concept. For example, vehicles can be divided into two classes, 1) Vehicles that run on the road and 2) Vehicles that run on the rail, but language may not have specific words to describe these classes.
- *Confusion between object level and meta level concept:* The synset abstraction seems to include both object-level concepts, such as Set, Time, and Space, and meta-level concepts, such as Attribute and Relation [7].
- *Heterogeneous levels of generality:* Two hyponyms of a concept may represent different level of generality. For example, as a hyponymy of concept 'aḍavi (forest)', there is a general concept 'podalu (bushes)' and a more specific concept 'kalabanda (aloe vera)', a medicinal plant. We are induced to consider the formers as types and the latter as roles. In other words, we discover that, if at first sight some synsets sound intuitively too specific when compared to their siblings, from a formal point of view, we may often explain their "different generality" by means of the distinction between types and roles [8].

2.3 Variants of WSD

WSD is broadly categorized into two types [3]:

- *Target Word WSD:* The target WSD system disambiguates a restricted set of target words, usually one per sentence. Supervised approaches are generally used for WSD where a tagged corpus is used to train the model. This trained model is then used to disambiguate the words in the target document.
- *All Word WSD:* The all word WSD system disambiguates all open-class words in the target document. Supervised approaches face the problem of data sparseness and it is not always possible to have a large tagged corpus for training. Hence, unsupervised methods are preferred in the case of all word WSD.

2.4 Approaches for Word Sense Disambiguation

WSD approaches are often classified according to the main source of knowledge used in sense differentiation. We are listing a few as discussed in [3].

- Supervised WSD Approaches: Supervised methods formulate WSD as a classification problem. The senses of a word represent classes and a classifier assigns a class to each new instance of a word. Any classifier from the machine learning literature can be applied. In addition to a dictionary, these algorithms need at least one annotated corpus, where each appearance of a word is tagged with the correct sense.
- Unsupervised WSD Approaches: Creating annotated corpus for all language-domain pairs is impracticable looking at the amount of time and money required. Unsupervised methods have the potential to overcome the new knowledge acquisition bottlenecK. These methods are able to induce word senses from training text by clustering word occurrences and then classifying new occurrences into the induced clusters/senses.
- Knowledge Based WSD Approaches: WSD heavily depends on knowledge and this knowledge must be in the machine readable format. There are various structures designed for this purpose like tagged and untagged corpora, machine-readable dictionaries, ontologies, etc. The main use of lexical resources in WSD is to associate senses with words. Here, selectional restrictions, overlap of definition text, and semantic similarity measures are used for knowledge based WSD.

3 Data Collection

Telugu is the second most spoken language in India. It is one of the twenty-two official languages of the Republic of India and the official language of the states of Telangana and Andhra Pradesh. Telugu has a vast and rich literature dating back to many centuries [9].

However, there is no generally accessible dictionary reference till date. In this work, a Telugu lexicon was created manually from 'శ్రీసూర్యరాయాంధ్ర తెలుగు నిఘంటువు (Srī sūryarāyāṃdhra Telugu nighaṃṭuvu)' which has 8 volumes in total [14]. Nearly 21,000 root words alongside their meanings were recorded as part of this paper.[3] The resource is developed to enrich OntoSenseNet[4] with addition of regional language resources. For each word extracted, based on its meaning, sense was identified by native speakers of language. We are presenting some statistics of available resources in Table 1. There are around 36,000 words in the dictionary we developed whereas IndoWordNet lists 21,091 words. Even without further analysis and classification we can see that this resource enriches WordNet by adding almost 15,000 words. This was the motivation to start this work. Nouns are still being added to our resource. Telugu-Hindi and English-Telugu dictionaries are available[5].

[3] https://github.com/Shreekavithaa/MS-Thesis-Files/tree/master/Data.
[4] http://ceh.iiit.ac.in/lexical_resource/index.html.
[5] https://ltrc.iiit.ac.in/onlineServices/Dictionaries/Dict_Frame.html.

Table 1. Statistics of available lexical resources for Telugu

Resource	Verbs	Adverbs	Adjectives
OntoSenseNet	8483	253	1673(In progress)
Telugu WordNet[a]	2803	477	5827
Synsets in WordNet	2795	442	5776
Telugu-Hindi Dictionary[b]	9939	142	1253
English-Telugu Dictionary[c]	4657	1893	6695

3.1 Validation of the Resource

Cohen's Kappa [4] was used to measure inter-annotator agreement which proves the reliability. The annotations are done by one human expert and it is cross-checked by another annotator who is equally proficient. Both the annotators are native speakers of the language. Verbs and adverbs are randomly selected from our resource for the evaluation sample. The inter-annotator agreement for 500 Telugu verbs is 0.86 and for 100 Telugu adverbs it is 0.94. Validation of the language resource shows high agreement [10]. However further validation of the resource is in progress.

4 Ontological Classification Used for Annotation

The formal ontology we used in this paper is proposed by [17]. This is based on various theories given in Indian grammatical tradition. The two main propositions given in Indian grammatical tradition are: (a) All words (noun and verb) in a language can be derived from verbal root (Sanskrit, dhātu). (b) Verbs have operation/process as its predominant element [17]. Theory used in this paper believes that meanings have primitive ontological forms and aims at extensive coverage of language.

4.1 Verb

Verbs are considered as the most important lexical and syntactic category of language. Verbs provide relational and semantic framework for its sentences. In a single verb many verbal sense-types can be present and different verbs may share same verbal sense-types. There are seven sense-types of verbs have been derived by collecting the fundamental verbs used to define other verbs [13]. These sense-types are inspired from different schools of Indian philosophies. The seven sense-types of verbs are listed below [16] with their primitive sense along with Telugu examples.

– Means|End - A process which cannot be accomplished without a doer (To do). Examples: *parugettu (run), moyu (carry)*

– Before|After - Every process has a movement in it. The movement maybe a change of state or location (To move). Examples: *pravāhaṁ (flow), oragupovu (lean)*
– Know|Known - Conceptualize, construct or transfer information between or within an animal (To know). Examples: *daryāptu (investigate), vivaraṅa (explain)*
– Locus|Located - Continuously having (to be in a state) or possessing a quality (To be). Examples: *Ādhārapaḍi (depend), kaṅgāru (confuse)*
– Part|Whole - Separation of a part from whole or joining of parts into a whole. Processes which causes a pain. Processes which disrupt the normal state (To cut). Examples: *perugu (grow), abhivṛddhi (develop)*
– Wrap|Wrapped - Processes which pertain to a certain specific object or category. It is like a bounding (To cover). Examples: *dhariñcaḍaṁ (wear), Āśrayaṁ(shelter)*
– Grip|Grasp - Possessing, obtaining or transferring a quality or object (To have). Examples: *lāgu (grab), vārasatvaṅga (inherit)*

4.2 Adverb

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 as reported by [13]. Sense-classes with explanation are illustrated with Telugu examples in Table 2.

Table 2. Sense-Class categorization of Adverbs

Sense-Class	Explanation	Example
Temporal	Adverbs that attributes to sense of time	akāraṇamu
Spatial	Adverbs that attributes to physical space	diguvagā
Force	Adverbs that attributes to cause of happening	nikkamu
Measure	Adverbs dealing with comparison	niṃḍu

4.3 Adjectives

Like verbs, adjectives are also collocative in nature. [13] identifies 12 sense-types. However these can be reduced to 6 pairs. Sense-Types of adjectives with explanation are illustrated with Telugu examples in Table 3.

4.3.1 Spatio-Temporal Classification of Adjectives

Additional information that we can use for classification are the locational and temporal attributes of adjectives. This can help the machine understand the sense in which a particular adjective is used. In this paper we are proposing three classes, namely, (a) Adjectives dealing with disposition (b) Adjectives of experience (c) Adjectives that talk about the behavior.

Table 3. Sense-Type classification of adjectives

Sense-Type	Explanation	Example
Locational	Adjectives that universalize or localize a noun	nirdista (specific)
Quantity	Adjectives that either qualify cardinal measure or quantify in ordinal-type	okkati (One)
Relational	Adjectives that qualify nouns in terms of dependence or dispersal	vistrta (broad)
Stress	Adjectives that intensify or emphasis a noun	gatti (strong)
Judgement	Adjectives that qualify evaluation or qualify valuation feature of a noun	mamci (good)
Property	Adjectives that attribute a nature or qualitative domain of a noun	nallani (black)

- *Disposition:* These are the tendencies we have and our habits. We do these unconsciously with not much thought. In ontological terms, this is the trans-temporal categorization.
  Example: mañci vyakti (A good person) Here mañci(good) is used to determine the quality of a man. Here his goodness is reflected in all of his doings. This is an opinion that could be formed after observing him over time and not by judging any one action. Hence it is trans-temporal.
- *Experience:* These could be defined as the adjectives which express the emotion or cognition at any particular moment in time.
  Example: kōpantō unna vyakti (an angry man). This shows the state of the man at that particular point in time.
- *Behavior:* This category of adjectives describe the physical attributes and bodily actions. Hence this is the spatial categorization.
  Example: biggaragā aravaḍaṁ (Loud scream). This describes the action of a person.

This classification is attempted for 400 adjectives in Telugu.

5 Annotation Procedure

Every verb, in OntoSenseNet, can have all the seven meaning primitives (sense-types) in it, in various degrees. The degree depends on the usage or popularity of a meaning in a language that leads to a particular sense-type annotation. In our resource we have identified two sense-types for each verb, i.e. primary and secondary. Entire lexicon of verbs and adverbs is classified. However, work is in progress for adjectives. Till date, 1673 adjectives are annotated. All of the annotations are done manually by native speakers of language in accordance with the classification presented in Sect. 4.

5.1 Enrichment of the Resource

We did not overlook the possibility of existence of unseen words in Telugu WordNet but not in our resource. Out of 2795 verb synsets, we extracted a bag of words which are not present in the resource in developed. We annotated each lexeme of these synsets as an separate entry. We followed similar annotation guidelines for the synsets of WordNet as well. Annotations for this set of lexemes are crowd-sourced and annotations are done following the annotation guidelines by six language experts. We can observe that all the lexemes in synsets (in Telugu wordNet) don't share the same primary sense-type. Another hypothesis is that having sets that share the same primary and secondary sense-types would result in better WSD for tasks like machine translation. However, this hypothesis needs further experimental validation.

5.1.1 Adding Synsets from WordNet to Our Resource

ID :: 3434
CAT :: verb
CONCEPT :: ప్రతిరోజు సూర్యుడు తూర్పున రావడం (pratiroju sūryuḍu tūrpuna rāvaḍaṃ)
EXAMPLE :: సూర్యుడు తూర్పున ఉదయిస్తాడు (sūryuḍu tūrpuna udayistāḍu)
SYNSET-TELUGU :: ఉదయించు (udayiṃcu), పుట్టు (puṭṭu), పొడతెంచు (poḍateṃcu), అవతరించు (avatariṃcu), ఆవిర్భవించు (āvirbhaviṃcu), ఉద్భవించు (udbhaviṃcu), జనించు (janiṃcu), జనియించు (janiyiṃcu), ప్రభవించు (prabhaviṃcu), వచ్చు (vaccu), ఏతెంచు (eteṃcu)

In synset ID 3434, the verb puṭṭu (birth) is used in the sense of Sun rising in the east. In a sense that sun is taking birth i.e. it conveys that sun came into existence. The primary sense of this would be 'Before|After' as it deals with transition. Secondary sense would be 'Locus|Located' as it shows the state of a sun in the dawn.

However, another (synonymous) word, janiṃcu (birth), in the synset is used to describe the birth of a child. In a sense that a mother gave birth to her child. This process of child-birth needs an agent hence the primary sense becomes 'Means|End' as the action needs agent for its accomplishment. The secondary sense would be 'Part|Whole' as the child was separated from a whole i.e. his mother.

In this example, words from same synset have different primary sense-type. There is a high potential for such occurrences hence each word in the synset was considered as a new entry for the annotation task rather than assigning same primary and secondary sense-type to all the words in a synset.

6 Challenges During Annotation

A lot of times during annotation, words with confusing sense-types have occurred. For example, consider the words : కోపించు (kopiṃcu); ఎదురుకొలుపు (edurukolupu)

Some words that are in the developed Telugu dictionary are not in use anymore. Many such forgotten words are encountered and even the gloss of the words weren't helpful for the annotation. Such words are left out. For example: అగడాడ (agaḍāḍa): ప్రేలు (prelu), వదరు (vadaru); దరికొను (darikonu) : దహించు (dahiṃcu) , కాల్చు (kālcu)

In some cases, the word is unknown however knowing the gloss helped in the classification task. For example: అక్కటికించు (akkaṭikiṃcu) : కరుణించు (karuṇiṃcu), జాలిపడు (jālipaḍu); అండగొట్టు (aṃḍagoṭṭu) : ఆలస్యము చేయు (ālasyamu ceyu); త్రస్తరించు (trastariṃcu) : క్రిందుపరుచు (kriṃduparucu), అధఃకరించు (adhaḥkariṃcu)

The annotations done involve a lot of time and manual labor hence they are very cost intensive.

7 Comparative Analysis

Similar resources have been developed for English and Hindi as well. The differences in the sense distribution of these languages could be due the syntactic and semantic properties of the language. Telugu has many inflections and it is highly agglutinative.

Figure 2 shows the sense-type distribution for English, Hindi and Telugu verbs in OntoSenseNet.

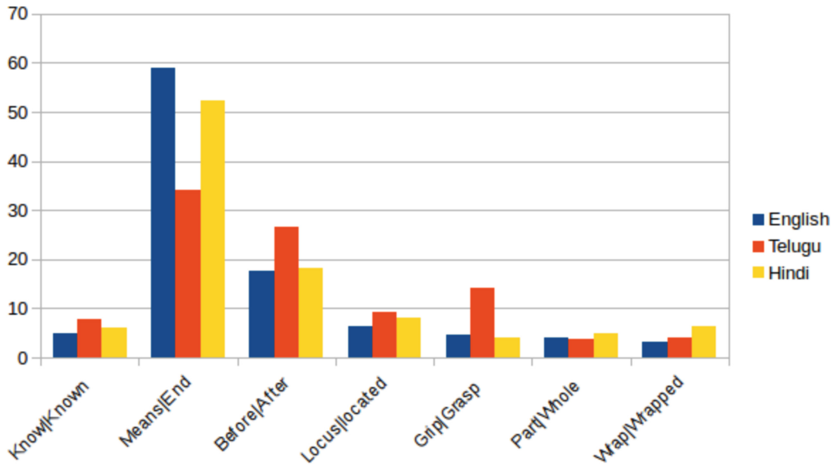


Fig. 2. Verb sense-type distribution across langauges

Table 4 shows sense-class distribution of adverbs for OntoSenseNet-English, OntoSenseNet-Hindi and OntoSenseNet-Telugu.

Table 4. 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%

8 Adverbial Class Distribution of Verbs

We have extracted all the <Verb, Adverb> and <Adverb, Verb> pairs from the Telugu Wikipedia. In order to acquire these patterns we performed the task of POS tagging[6] on Wikipedia corpus. From the extracted pairs, we noticed that there are comparatively more <Adverb, Verb> pairs than <Verb, Adverb> pairs which align with the structure of Telugu language [18]. 400 verbs and 445 adverbs are annotated according to the formal ontology that is discussed in Sect. 4, A Formal Ontology-based Classification of Lexemes. These words formed about 2000 <Verb, Adverb> and <Adverb, Verb> pairs. Our aim is to study the adverbial class distribution of verbs in Telugu. [13] proves that such annotations help in disambiguating the word senses thus result in improved word-sense disambiguation (WSD) task(s). This is one of the major applications of OntoSenseNet.

In Table 5, we show the adverbial class distribution of verbs in <Verb, Adverb> and <Adverb, Verb> pairs. Adverbial sense-classes are labeled as columns and sense-types of verbs are labeled as rows. Any cell in the table represents the percentage of a 'sense-class' of adverbs that modify a particular 'sense-type' of verbs.

Column-1 of Table 5 means 20.0% of 'spatial'; 13.6% of 'temporal'; 18.8% of 'force' and 24.4% of 'measure' sense-classed of adverbs modify 'to know' sense-type of verbs. This shows that majority of the 'to know' verbs are primarily modified by adverbs with 'measure' sense-class. For example : *cālā anipiṃciṃdi* (feel immensely). 'To move', 'to do' verbs are primarily modified by 'spatial', 'force' sense-class of adverbs respectively. Examples are *nerugā māṭlāḍutāḍu*(talk in a straight forward manner), *emoṣanalgā ālocistāḍu* (think emotionally). 'Temporal' sense-class of adverbs can modify all the sense-types of verbs. 'To be' sense-type of verbs is also significantly modified by 'force' sense-class of adverbs. However, 'temporal' and 'measure' sense-classes also seem to show comparable performance in modifying 'to be' sense-type. We can find many such examples in Telugu language.

[6] https://bitbucket.org/sivareddyg/telugu-part-of-speech-tagger.

Table 5. Adverb Sense-Class Distribution in <Verb,Adverb> pairs

	To Know	To Move	To Do	To Have	To Be	To Cut	To Bound
Spatial	20.0%	28.5%	20%	9.5%	9.5%	8.5%	4.0%
Temporal	13.6%	22.0%	14.6%	20.5%	20.5%	4.4%	4.4%
Force	18.8%	21.5%	22.2%	7.2%	22.9%	6.0%	4.1%
Measure	24.4%	16.5%	19.5%	5.2%	20.3%	7.5%	3.8%

9 Conclusion

In this paper, a manually annotated sense lexicon was developed. Classification was done by expert native Telugu speakers. This sense-annotated resource is an attempt to make machine as intelligible as a human while performing WSD tasks. Hence, without limiting to the word and its meaning, we attempted to convey the sense in which humans understand a sentence. The validation of this resource was done using Cohen's Kappa that showed higher agreement. Further validation and enrichment of the resource is in progress. Classification of WordNet was attempted to see if the proposed classification could improve WSD tasks. All the data mentioned in this paper can be found on Github.[7]

10 Future Work

Annotations of all available synsets of the WordNet needs to be done to spot the anomalies. The anomalies should be studied to further enrich Telugu WordNet. Tagging of adjectives in still in progress. Supervised WSD approaches are to be implemented by using the OntoSenseNet, sense-annotated corpora. Knowledge based WSD approaches can also benefit largely from such sense-annotated corpora hence such classifiers could be implemented as well. We need to measure the significance of this lexicon in NLP tasks. There are several ontological problems with the WordNet and we are attempting to solve a part of those with the proposed formal ontology. But a far more important question is "How can we know when an ontology is complete?". We hope to arrive at an answer for this question in near future.

Acknowledgements. This work is part of the ongoing MS thesis in Exact Humanities under the guidance of Prof. Navjyoti Singh. I am immensely grateful to my mother Mrs. Vijaya Lakshmi Kiran Kumar for helping me with data collection and annotation process of the whole resource. I would also like to show my gratitude to K Jithendra Babu, Historian & Chairman at Deccan Archaeological and Cultural Research Institute, Hyderabad for providing the hard copy of the dictionary (all the 8 volumes most of which are unavailable currently) and for sharing his pearls of wisdom with us during the course of this research. I thank my fellow researchers from LTRC lab, IIIT-H who provided insight and expertise that greatly assisted the research.

[7] https://github.com/Shreekavithaa/MS-Thesis-Files/tree/master/Data.

References

1. Bhatt, B., Bhattacharyya, P.: Indowordnet and its linking with ontology. In: Proceedings of the 9th International Conference on Natural Language Processing (ICON-2011). Citeseer (2011)
2. Bhattacharyya, P.: Indowordnet. lexical resources engineering conference 2010 (LREC 2010). Malta, May 2010
3. Bhingardive, S., Bhattacharyya, P.: Word sense disambiguation using IndoWordNet. In: Dash, N.S., Bhattacharyya, P., Pawar, J.D. (eds.) The WordNet in Indian Languages, pp. 243–260. Springer, Singapore (2017). https://doi.org/10.1007/978-981-10-1909-8_15
4. Carletta, J.: Assessing agreement on classification tasks: the kappa statistic. Comput. Linguist. **22**(2), 249–254 (1996)
5. Chatterjee, A., Joshi, S.R., Khapra, M.M., Bhattacharyya, P.: Introduction to tools for indowordnet and word sense disambiguation. In: 3rd IndoWordNet workshop, International Conference on Natural Language Procesing (2010)
6. Gamut, L.: Logic, Language, and Meaning, volume 1: Introduction to Logic, vol. 1. University of Chicago Press (1991)
7. Gangemi, A., Guarino, N., Masolo, C., Oltramari, A.: Sweetening wordnet with dolce. AI Mag. **24**(3), 13 (2003)
8. Gangemi, A., Guarino, N., Oltramari, A.: Conceptual analysis of lexical taxonomies: The case of wordnet top-level. In: Proceedings of the International Conference on Formal Ontology in Information Systems-Volume 2001, pp. 285–296. ACM (2001)
9. Kumar, M.S., Murthy, K.N.: Automatic construction of telugu thesaurus from available lexical resources
10. Landis, J.R., Koch, G.G.: The measurement of observer agreement for categorical data. Biometrics **33**, 159–174 (1977)
11. Niles, I., Pease, A.: Towards a standard upper ontology. In: Proceedings of the International Conference on Formal Ontology in Information Systems-Volume 2001, pp. 2–9. ACM (2001)
12. Niles, I., Pease, A.: Linking lixicons and ontologies: mapping wordnet to the suggested upper merged ontology. In: Ike, pp. 412–416 (2003)
13. Otra, S.: towards building a lexical ontology resource based on intrinsic senses of words. Ph.D. thesis, International Institute of Information Technology Hyderabad (2015)
14. Pantulu, J.: Sri Suryaraayandhra Telugu Nighantuvu, vol. 1–8. Telugu University (1988)
15. Pease, A., Fellbaum, C., Vossen, P.: Building the global wordnet grid. CIL18 (2008)
16. Rajan, K.: Understanding verbs based on overlapping verbs senses. In: 51st Annual Meeting of the Association for Computational Linguistics Proceedings of the Student Research Workshop, pp. 59–66 (2013)
17. Rajan, K.: Ontological classification of verbs based on overlapping verb senses (2015)
18. RJ, R.S., KV, M.M., et al.: Assessment and development of POS tag set for Telugu. In: Proceedings of the 6th Workshop on Asian Language Resources (2008)
19. Sravanthi, M.C., Prathyusha, K., Mamidi, R.: A dialogue system for Telugu, a resource-poor language. In: Gelbukh, A. (ed.) CICLing 2015. LNCS, vol. 9042, pp. 364–374. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-18117-2_27

Machine Translation

A Neural Network Classifier Based on Dependency Tree for English-Vietnamese Statistical Machine Translation

Viet Hong Tran[1,2(✉)], Quan Hoang Nguyen[2(✉)], and Vinh Van Nguyen[2(✉)]

[1] University of Economic and Technical Industries, Hanoi, Vietnam
thviet@uneti.edu.vn
[2] University of Engineering and Technology, Vietnam National University, Hanoi, Vietnam
quan94fm@gmail.com, vinhnv@vnu.edu.vn

Abstract. Reordering in MT is a major challenge when translating between languages with different of sentence structures. In Phrase-based statistical machine translation (PBSMT) systems, syntactic pre-ordering is a commonly used preprocessing technique. This technique can be used to adjust the syntax of the source language to that of the target language by changing the word order of a source sentence prior to translation and solving to overcome a weakness of classical phrase-based translation systems: long distance reordering. In this paper, we propose a new pre-ordering approach by defining dependency-based features and using a neural network classifier for reordering the words in the source sentence into the same order in target sentence. Experiments on English-Vietnamese machine translation showed that our approach yielded a statistically significant improvement compared to our prior baseline phrase-based SMT system.

Keywords: Natural language processing · Machine translation · Phrase-Based statistical machine translation · Pre-ordering · Dependency tree

1 Introduction

Recently the phrase-based and neural-based become dominant methods in current machine translation. Statistical machine translation (SMT) systems achieved a high performance in many typologically diverse language pairs. In phrase-based statistical machine translation (PBSMT) [1,2], syntactic pre-ordering is a commonly used preprocessing technique. It adjust the syntax of the source language to that of the target language by changing the word order of the source sentence prior to translation. This technology can overcome a weakness of classical phrase-based translation systems: long distance reordering. This is a major source of errors when translating between languages with difference of sentence structures. Phrase-based translation systems do not place a similar prior penalty on phrase reordering during decoding, however, such systems have been shown to profit from syntactic pre-ordering as well.

Many solutions to the reordering problem have been proposed, such as syntax-based model [3], lexicalized reordering [2], and tree-to-string methods [4]. Chiang [3] shows significant improvement by keeping the strengths of phrases, while incorporating syntax into SMT. Some approaches have been applied at the word-level [5]. They are particularly useful for language with rich morphology, for reducing data sparseness. Other kinds of syntax reordering methods require parser trees, such as the work in [5,6]. The parsed tree is more powerful in capturing the sentence structure. However, it is expensive to create tree structure, and building a good quality parser is also a hard task. All the above approaches require much decoding time, which is expensive.

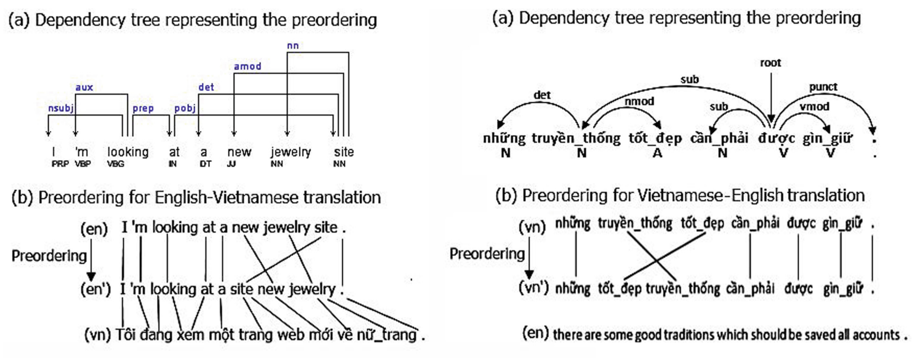

(a) Dependency tree representing the preordering
nsubj
aux
prep
pobj
det
amod
nn
I 'm looking at a new jewelry site
PRP VBP VBG IN DT JJ NN NN
(b) Preordering for English-Vietnamese translation
(en) I 'm looking at a new jewelry site .
Preordering
(en') I 'm looking at a site new jewelry .
(vn) Tôi đang xem một trang web mới về nữ_trang .
(a) Dependency tree representing the preordering
det
nmod
sub
sub
root
punct
vmod
những truyền_thống tốt_đẹp cần_phải được gìn_giữ .
N N A N V V .
(b) Preordering for Vietnamese-English translation
(vn) những truyền_thống tốt_đẹp cần_phải được gìn_giữ .
Preordering
(vn') những tốt_đẹp truyền_thống cần_phải được gìn_giữ .
(en) there are some good traditions which should be saved all accounts .

Fig. 1. Example of preordering for English-Vietnamese translation and Vietnamese-English translation.

The end-to-end neural MT (NMT) approach [7] has recently been proposed for MT. The NMT system usually causes a serious out-of-vocabulary (OOV) problem, the translation quality would be badly affected ; The NMT decoder lacks a mechanism to guarantee that all the source words are translated and usually favors short translations. It is difficult for an NMT system to benefit from target language model trained on target monolingual corpus, which is proven to be useful for improving translation quality in statistical machine translation (SMT). NMT need much more training time. In [8], NMT requires longer time to train (18 days) compared to their best SMT system (3 days)

The approach we are interested in here is to balance the quality of translation with decoding time. Reordering approaches as a preprocessing step [9–12] are very effective (significant improvement over state of-the-art phrase-based and hierarchical machine translation systems and separately quality evaluation of each reordering models).

Inspired by this preprocessing approaches, we propose a combined approach which preserves the strength of phrase-based SMT in reordering and decoding time as well as the strength of integrating syntactic information in reordering. Firstly, the proposed method uses a dependency parsing for preprocessing step with training and testing. Secondly, transformation rules are applied to reorder the source sentences. The experimental resulting from English-Vietnamese pair shows that our approach achieved improvements in BLEU scores [13] compared to MOSES [14] which is the state of-the-art phrase-based SMT system.

This paper is structured as follows: Sect. 1 introduces the reordering problem, Sect. 2 reviews the related works. Section 3 briefly introduces classifier-based neural network Preordering for Phrase-based SMT. Section 4 describes experimental results. Section 5 discusses the experimental results. And, conclusions are given in Sect. 6.

2 Related Works

The difference of the word order between source and target languages is the major problem in phrase-based statistical machine translation. Fig. 1 describes an example that a reordering approach modifies the word order of an input sentence of a source languages (English) in order to generate the word order of a target languages (Vietnamese).

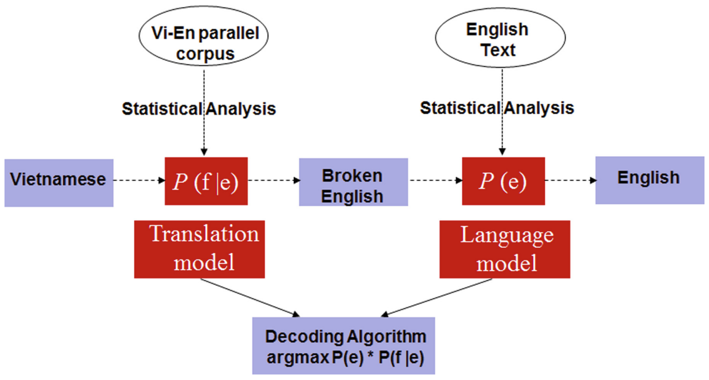


Fig. 2. An example phrase-based statistical machine translation in Moses toolkit.

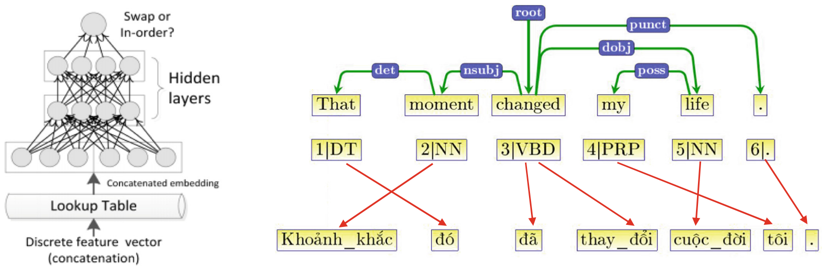
Many preordering methods using syntactic information have been proposed to solve the reordering problem. (Collin 2005; Xu 2009) [5, 10] presented a preordering method which used manually created rules on parse trees. In addition, linguistic knowledge for a language pair is necessary to create such rules. Other preordering methods using automatic created reordering rules or a statistical classifier were studied [12, 15]

Collins [5] developed a clause detection and used some handwritten rules to reorder words in the clause. Partly, (Habash 2007)[16] built an automatic extracted syntactic rules. Xu [10] described a method using a dependency parse tree and a flexible rule to perform the reordering of subject, object, etc... These rules were written by hand, but [10] showed that an automatic rule learner can be used.

Bach [17] propose a novel source-side dependency tree reordering model for statistical machine translation, in which subtree movements and constraints are represented as reordering events associated with the widely used lexicalized reordering models.

(Genzel 2010; Lerner and Petrov 2013) [11, 12] described a method using discriminative classifiers to directly predict the final word order. Cai [18] introduced a novel pre-ordering approach based on dependency parsing for Chinese-English SMT.

Isao Goto [19] described a preordering method using a target-language parser via cross-language syntactic projection for statistical machine translation.



(a) (b)

Pair	Head			Left child			Right child			distance	Punctuation	Label
	x_h	$T(x_h)$	$L(x_h)$	x_c	$T(x_c)$	$L(x_c)$	x_c	$T(x_c)$	$L(x_c)$	$d(x_h,x_c)$	$w(x_h,x_c)$	
(3,2)	changed	VBD	root	moment	NN	nsubj	Null	Null	Null	-1	0	0
(3,5)	changed	VBD	root	Null	Null	Null	life	NN	dobj	+1	0	0
(2,1)	moment	NN	nsubj	That	DT	det	Null	Null	Null	-1	0	1
(5,4)	life	NN	dobj	my	PRP	poss	Null	Null	Null	-1	0	1

(c)

Pair	Left child				Right child				Head		Punctuation	Label
	x_l	$T(x_l)$	$L(x_l)$	$d(x_h, x_l)$	x_r	$T(x_r)$	$L(x_r)$	$d(x_h, x_r)$	x_h	$T(x_h)$	$w(x_l,x_r)$	
(2,5)	moment	NN	nsubj	-1	life	V	dobj	+1	changed	VBD	0	0

(d)

Fig. 3. A reordering model for statistical machine translation: (a) neural network classifier architecture; (b) an aligned English-Vietnamese parallel sentence pair with sample extracted training instances and features for (c) head-child classifier and (d) sibling classifier.

Joachim Daiber [20] presented a novel examining the relationship between pre-ordering and word order freedom in Machine Translation.

Chenchen Ding, [21] proposed extra-chunk pre-ordering of morphemes which allows Japanese functional morphemes to move across chunk boundaries.

Christian Hadiwinoto presented a novel reordering approach utilizing sparse features based on dependency word pairs [22] and presented a novel reordering approach utilizing a neural network and dependency-based embedding to predict whether the translations of two source words linked by a dependency relation should remain in the same order or should be swapped in the translated sentence [8]. This approach is complex and spend much time to process.

Our approach is closest similarity to [12], [8] but it has a few differences. Firstly, we aimed to develop the phrase-based translation model using dependency parse of source sentence to translate from English to Vietnamese. Secondly, we extracted automatically a set of English to Vietnamese transformation rules from English-Vietnamese parallel corpus by using Neural Network classification model with lexical and syntactic features based on dependency parsing of source sentence. Thirdly, we use the neural network classifier to build two models that directly predict target-side word as a preprocessing step in phrase-based machine translation. As the same with [9,16], we also applied preprocessing in both training and decoding time.

3 A Neural Network Classifier-Based Preordering for Phrase-Based SMT

3.1 Phrase-Based SMT

In this section, we will describe the phrase-based SMT system which was used for the experiments. Phrase-based SMT, as described by [1] translates a source sentence into a target sentence by decomposing the source sentence into a sequence of source phrases, which can be any contiguous sequences of words (or tokens treated as words) in the source sentence. For each source phrase, a target phrase translation is selected, and the target phrases are arranged in some order to produce the target sentence. A set of possible translation candidates created in this way were scored according to a weighted linear combination of feature values, and the highest scoring translation candidate was selected as the translation of the source sentence. Symbolically,

$$\hat{t} = {}_{t,a} \sum_{i=1}^{n} \lambda_i f_j(s, t, a) \tag{1}$$

Feature	Description
Pair	Pair word with head-child relation
x_h	The head word x_h
$T(x_h)$	Part-of-speech (POS) tag of x_h
$L(x_h)$	The dependency label $L(x_h)$ linking x_h to head word of x_h
x_{cl}	The child word x_c if child left
$T(x_{cl})$	Part-of-speech (POS) tag of x_{cl}
$L(x_{cl})$	The dependency label L(xh) linking x_h to x_h
x_{cr}	The child word x_c if child right
$T(x_{cr})$	Part-of-speech (POS) tag of x_{cr}
$L(x_{cr})$	The dependency label $L(x_h)$ linking x_h to x_h
$d(x_h, x_c)$	The signed distance between the head and the child in the original source sentence: -2 if x_{cl} is on the left of x_h and there is at least one other child between them – 1 if x_{cl} is on the left of x_h and there is no other child between them +1 if x_{cr} is on the right of x_h and there is no other child between them + 2 if x_{cr} is on the right of x_h and there is no other child between them
ω (x_h, x_c)	A Boolean $\omega(x_h, x_c)$ to indicate if any punctuation symbol, which is also the child of x_h, exists between x_h and x_c
Label	The label 1 or 0 indicates whether the two words need to be swapped or kept in order

(a) The feature of Head-child classifier

Feature	Description
Pair	Pair word with head-child relation
x_l	The left child word x_l
$T(x_l)$	Part-of-speech (POS) tag of x_l
$L(x_l)$	The dependency label $L(x_l)$ linking x_l to x_h
$d(x_h,x_l)$	the signed distance x_l to its head x_h +1 if x_{cr} is on the right of x_h and there is no other child between them + 2 if x_{cr} is on the right of x_h and there is no other child between them
x_r	The right child word x_r
$T(x_r)$	Part-of-speech (POS) tag of x_r
$L(x_r)$	The dependency label $L(x_r)$ linking x_r to x_h
$d(x_h,x_r)$	the signed distance x_r to its head x_h: -2 if x_{cl} is on the left of x_h and there is at least one other child between them – 1 if x_{cl} is on the left of x_h and there is no other child between them
x_h	The head word x_h
$T(x_h)$	Part-of-speech (POS) tag of x_h
$\omega(x_l, x_r)$	A Boolean $\omega(x_l, x_r)$ to indicate if any punctuation symbol, which is also the child of x_h, exists between x_l and x_r
Label	The label 1 or 0 indicates whether the two words need to be swapped or kept in order

(b) The feature of sibling classifier

Fig. 4. (a) The feature of head-child relation and (b) the feature of sibling relation used in training data from corpus English-Vietnamese

when s is the input sentence, t is a possible output sentence, and a is a phrasal alignment that specifies how t is constructed from s, and $\hat{t}$ is the selected output sentence. The weights λ_i associated with each feature f_i are tuned to maximize the quality of the translation hypothesis selected by the decoding procedure that computes the argmax.

The log-linear model is a natural framework to integrate many features. The probabilities of source phrase given target phrases, and target phrases given source phrases, are estimated from the bilingual corpus.

[1] used the following distortion model (reordering model), which simply penalizes nonmonotonic phrase alignment based on the word distance of successively translated source phrases with an appropriate value for the parameter α:

$$d(a_i - b_{i-1}) = \alpha^{|a_i - b_{i-1} - 1|} \tag{2}$$

Current time, state-of-the-art phrase-based SMT system using the lexicalized reordering model in Moses toolkit. In our work, we also used Moses to evaluate on English-Vietnamese machine translation tasks. Figure 2 show an architecture of Phrase-based Statistical Machine Translation in Moses toolkit.

3.2 Classifier-Based Preordering

In this section, we describe the learning model that can transform the word order of an input sentence to an order that is natural in the target language. English is used as source language, while Vietnamese is used as target language in our discussion about the word orders.

For example, when translating the English sentence:

$$That\ moment\ changed\ my\ life.$$

to Vietnamese, we would like to reorder it as:

moment that changed life my.

And then, this model will be used in combination with translation model.

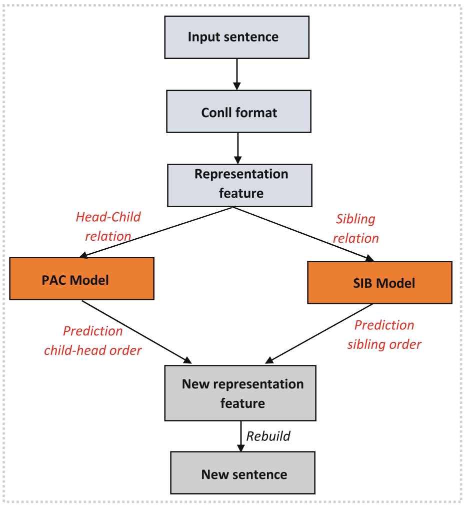


Fig. 5. Framework for preordering a new source sentence from parallel corpus.

Training Data for Preordering and Features. We use the dependency grammars and the differences of word order between English and Vietnamese to create a set of the reordering rules. For each source sentence, we conducted to extract the POS tags and head modifier dependencies correspond shown in Fig. 3. By traversing the dependency tree starting at the root to reordering, we determine the order of the head and its children for each head word and continue the traversal recursively in that order. In the above example, we need to decide the order of the head "changed" with the children "moment", "life"; the head "moment" with child "that", the head "life" with child "my".

The words in sentence are reordered by a new sequence learned from training data using two neural classifiers. The head-child classifier predicts the order of the translated words of a source word and its head word. The sibling classifier predicts the order of the translated words of two source words that both have the common head word.

The features extracted based on dependency tree and alignment information. We traverse the tree from the top, with each head-child and sibling relation we decide swap or no swap in dependency trees.

Classification Model. We train two classifiers with a head-child relation and with a sibling relation. Each binary classifier takes a set of features related to the two source words as its input and predicts if the translated words should be swapped (positive) or remain in order (negative) each number of possible children. In hence, the classifiers learn to trade off between a rich set of overlapping features. List of features are given in Fig. 4.

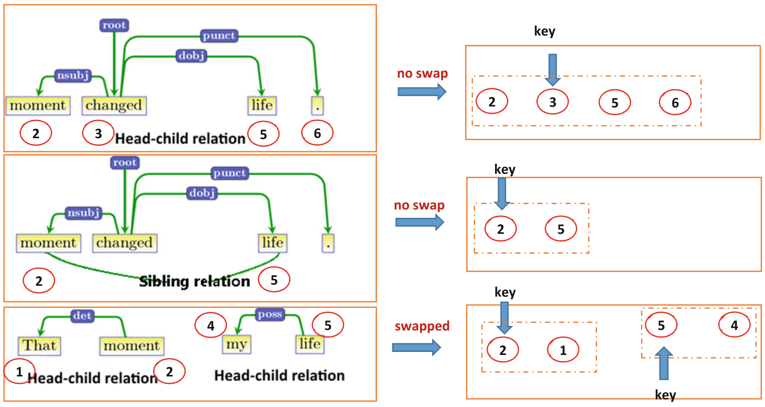


Fig. 6. An example for reordering after applying method classifier.

The classifier is a feed-forward neural network whose input layer contains the features. Each feature is mapped by a lookup table to a continuous vector representation. The resulting vectors are concatenated and fed into a series of hidden layers using the rectified linear activation function. Inspired from [8], we also initialize the hidden layers and the embedding layer for non-word features (POS tags, dependency labels, and

Boolean indicators) by a random uniform distribution. For word features x_h, x_c, x_l, and x_r, we initialize their embeddings by the dependency-driven embedding scheme of (Bansal, Gimpel, and Livescu 2014) [23]. This scheme is a modified skip-gram model, which given an input word, predicts its context, resulting in a mapping such that words with similar surrounding words have similar continuous vector representations (Mikolov et al. 2013) [24].

The training instances for the neural network classifiers are obtained from a word-aligned parallel corpus with head-child or sibling relation are extracted from their corresponding order label, swapped or in order, depending on the positions of their aligned target-side words. The NN classifiers are trained using back-propagation to minimize the cross-entropy objective function.

The learning algorithm produces a sparse set of features. In our experiments the our models have typically only a few 130K non-zero feature weights English-Vietnamese language pairs.

When extracting the features, every word can be represented by its word identity, its POS-tags from the treebank, syntactic label. We also include pairs of these features, resulting in potentially bilexical features.

We describe a method to build training data for a pair English to Vietnamese. Our purpose is to reconstruct the word order of input sentence to an order that is arranged as Vietnamese words order. For example with the English sentence in Fig. 3, after applying our framework in Fig. 5 for prediction two relation (head-child relation, sibling relation) and reordering as described in Fig. 6, the input sentence:

$$That\ moment\ changed\ my\ life.$$

Algorithm 1 Build Models

```
input: dependency trees of source sentences
          and alignment pairs;
output: Two neural network classifier model:
          - PAC Model (Head-child relation Model)
          - SIB Model (Sibling relation Model)
for each head-child relation pair in dependency trees of subset
and alignment pairs of sentences do
   generate PAC_feature (head-child relation + label) ;
for each sibling relation pair in dependency trees of subset
and alignment pairs of sentences do
   generate SIB_feature (sibling relation + label) ;
end for
Build PAC model from set of PAC_features;
Build SIB model from set of PAC_features;
```

Algorithm 2 Reordering

```
input: a source sentence;
output: a new source sentence;
for each dependency tree of a source sentence do
    for each head-child relation in tree do
        prediction head-child order from PAC Model
    end for
    for each sibling relation in tree do
        prediction sibling order from SIB Model
    end for
end for
Build new sentence;
```

is transformed into Vietnamese order:

$$moment\ that\ changed\ life\ my.$$

For this approach, we first do preprocessing to encode some special words and parser the sentences to dependency tree using Stanford Parser [25]. Then, we use target to source alignment and dependency tree to generate features. We add the information of the dependency tree as described in Fig. 4 with each relation (head-child relation and sibling relation) from the dependency tree. For each family in the tree, we generate a training instance if it has less than and equal four children.

For every node in the dependency tree, from the top-down, we find the node matching against the pattern in classifier model, and if a match is found, the associated order applied. We arrange the words in the English sentence, which is covered by the matching node, like Vietnamese words order. And then, we do the same for each children of this node.

The our algorithm's outline is given as Algorithm 1 and Algorithm 2

Algorithm 1 extract features and build models with input including dependency trees of source sentences and alignment pairs.

Algorithm 2 prediction order by considering head-child and sibling relation after finish Algorithm 1 from source-side dependency trees to build new sentence.

Table 1. Corpus statistical

Corpus	Sentence pairs	Training Set	Development Set	Test Set
General	133403	131019	1304	1080
			English	Vietnamese
Training	Sentences		131019	
	Average Length		19.34	18.09
	Word		2534498	2370126
	Vocabulary		50118	56994
Development	Sentences		1304	
	Average Length		18.19	17.13
	Word		28773	27101
	Vocabulary		3713	3958
Test	Sentences		1080	
	Average Length		21.5	20.9
	Word		28036	27264
	Vocabulary		3918	4316

Table 2. Our experimental systems on English-Vietnamese parallel corpus

Name	Description
Baseline	Phrase-based system
Baseline NMT	Neural machine translation (seq2seq)
Auto rules	Phrase-based system with corpus which is be preprocessing using automatic rules
Our method	Phrase-based system with corpus which is be preprocessing using neural network Classifier

The reordering decisions are made by two classifiers (head-child classifier and sibling classifier) where class labels correspond to decide swapped or no swapped. We train a separate classifier for each relation. Crucially, we do not learn explicit tree transformations rules, but let the classifiers learn to trade off between a rich set of overlapping features. To build a classification model, we use neural network classification model in the Tensorflow tools [26].

We apply them in a dependency tree recursively starting from the root node. If the POS-tags of a node matches the left-hand-side of the rule, the rule is applied and the order of the sentence is changed. We go through all the children of the node and matching rules for them from the set of automatically rules.

Figure 5 gives framework of original and process phrase in English. After apply this framework, with the source sentence in English: “that moment changed my life.”, and the target Vietnamese reordering Khoảnh_khắc đó đã thay_đổi cuộc_đời tôi . This sentences is arranged as the Vietnamese order. Vietnamese sentences are the output of our method. As you can see, after reordering, the original English has the same word order: “moment that changed life my.” in Fig. 1.

4 Experiment

In this section, we present our experiments to translate from English to Vietnamese in a statistical machine translation system. The language pair chosen is English-Vietnamese. We used Stanford Parser [25] to parse source sentence (English sentences).

We used dependency parsing and rules extracted from training the features-rich discriminative classifiers for reordering source-side sentences. The rules are automatically extracted from English-Vietnamese parallel corpus and the dependency parser of English examples. Finally, they used these rules to reorder source sentences. We evaluated our approach on English-Vietnamese machine translation tasks with systems in Table 2 which shows that it can outperform the baseline phrase-based SMT system.

We give some definitions for our experiments:

– Baseline: use the baseline phrase-based SMT system using the lexicalized reordering model in Moses toolkit.
– Baseline NMT: The state of the art NMT system with Tensorflow [27] was utilized. We used open source toolkits NMT in the TensorFlow seq2seq.

– Auto Rules : the phrase-based SMT systems applying automatic rules.
– Our method: the Phrase-based system with corpus which is preprocessed using neural network Classifier.

4.1 Implementation

– We used Stanford Parser [25] to parse source sentence and apply to preprocessing source sentences (English sentences).
– We used neural network classifier in Tensorflow tools [26] for training the features-rich discriminative classifiers to build model and apply them for reordering words in English sentences according to Vietnamese word order.
– We implemented preprocessing step during both training and decoding time.
– Using the SMT Moses decoder [14] for decoding.
– Using Pre-trained word vector [28] and dependency-driven continuous word representation [23] for the neural network classifiers.

4.2 Data Set and Experimental Setup

We used an English-Vietnamese corpus [29], including about 131019 pairs for training, 1080 pairs for testing and 1304 pairs for development test set. Table 1 gives more statistical information about our corpora. We conducted some experiments with SMT Moses Decoder [14] and SRILM [30]. We trained a trigram language model using interpolate and kndiscount smoothing with Vietnamese mono corpus. Before extracting phrase table, we use GIZA++ [31] to build word alignment with grow-diag-final-and algorithm. Besides using preprocessing, we also used default reordering model in Moses Decoder: using word-based extraction (wbe), splitting type of reordering orientation to three classes (monotone, swap and discontinuous – msd), combining backward and forward direction (bidirectional) and modeling base on both source and target language (fe) [14]. To contrast, we tried preprocessing the source sentence with manual rules and automatically rules.

Table 3. Translation performance for the English-Vietnamese task

System	BLEU (%)
Baseline	26.52
Baseline NMT	26.61
Auto Rules	27.05
Our method	27.17

4.3 BLEU Score

The result of experiments in Table 3 show our method to process the source sentences. In this method, we can find out various phrases in the translation model. So that, they enable us to have more options for decoder to generate the best translation.

Table 3 describes the BLEU score of our experiments. As we can see, by applying preprocessing in both training and decoding, the BLEU score of our best system increase by 0.26 point over "Baseline system". Improvement over 0.26 BLEU point is valuable because baseline system is the strong phrase based SMT (integrating lexicalized reordering models). We also carried out the experiments with Automatic rules [32]. Using automatic rules help the phrased translation model generate some best translation. Besides, by applying two models to prediction right order in each relation: head-child relation and sibling relation, we propose a new preordering approach which there is no rules based in our framework. The result proved that the effect of applying our method on the dependency tree when the BLEU score is higher than baseline systems.

5 Analysis and Discussion

We have found that in our experiments work is sufficiently correlated to the translation quality done manually. Besides, we also have found some error causes such as parse tree source sentence quality, word alignment quality and quality of corpus. All the above errors can effect reordering in translation system.

We focus mainly on explore the rich dependency feature combine input representation with word-embedding to build two models: PAC model for head-child relation and SIB model for sibling relation based on neural network classifier. Our study employed dependency syntactic and applying these models to reorder the source sentence and applied to English to Vietnamese translation systems.

Based on these phenomena, translation quality has significantly improved. We carried out error analysis sentences and compared to the golden reordering. Our analysis has also the benefits of our method on translation quality. In combination with machine learning method in related work [12], it is shown that applying classifier method to solve reordering problems automatically.

6 Conclusion

In this study, we propose a new pre-ordering approach for English-Vietnamese Statistical Machine Translation by defining dependency-based features and using a neural network classifier for reordering the words in the source sentence into the same order in target sentence. We used a neural network classifier in Tensorflow for training the features-rich discriminative classifiers and reordering words in English sentence according to Vietnamese word order.

We evaluated our approach on English-Vietnamese machine translation tasks. Experiments on English-Vietnamese machine translation show that our approach yields a statistically significant improvement compared to our prior baseline phrase-based SMT system. The experimental results showed that our approach achieved statistical improvements over a state-of-the-art phrase-based baseline system by BLEU point scores. We believe that such reordering rules benefit English-Vietnamese language pairs.

In the future, we plan to further investigate in this direction and use our method on other language pairs. We also attempt to create more efficient preordering rules by exploiting the rich information in dependency structures.

Acknowledgements. This work described in this paper has been partially funded by TC.02-2016-03 project: Building a machine translation system to support translation of documents between Vietnamese and Japanese to help managers and businesses in Hanoi approach Japanese market.

References

1. Koehn, P., Och, F.J., Marcu, D.: Statistical phrase-based translation. In: Proceedings of HLT-NAACL 2003, Edmonton, Canada, pp. 127–133 (2003)
2. Och, F.J., Ney, H.: The alignment template approach to statistical machine translation. Comput. Linguist. **30**(4), 417–449 (2004)
3. Chiang, D.: A hierarchical phrase-based model for statistical machine translation. In: Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005), Ann Arbor, Michigan, pp. 263–270, June 2005
4. Zhang, Y., Zens, R., Ney, H.: Chunk-level reordering of source language sentences with automatically learned rules for statistical machine translation. In: Proceedings of SSST, NAACL-HLT 2007/AMTA Workshop on Syntax and Structure in Statistical Translation, pp. 1–8 (2007)
5. Collins, M., Koehn, P., Kucerová, I.: Clause restructuring for statistical machine translation. In: Proceedings of ACL 2005, Ann Arbor, USA, pp. 531–540 (2005)
6. Quirk, C., Menezes, A., Cherry, C.: Dependency treelet translation: syntactically informed phrasal SMT. In: Proceedings of ACL 2005, Ann Arbor, Michigan, USA, pp. 271–279 (2005)
7. Wu, Y., et al.: Googleś neural machine translation system: bridging the gap between human and machine translation. arXiv preprint arXiv:1609.08144 (2016)
8. Hadiwinoto, C., Ng, H.T.: A dependency-based neural reordering model for statistical machine translation. arXiv preprint arXiv:1702.04510 (2017)
9. Xia, F., McCord, M.: Improving a statistical MT system with automatically learned rewrite patterns. In: Proceedings of Coling 2004, Geneva, Switzerland, COLING, pp. 508–514, 23–27 August 2004
10. Xu, P., Kang, J., Ringgaard, M., Och, F.: Using a dependency parser to improve SMT for subject-object-verb languages. In: Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Boulder, Colorado, pp. 245–253. Association for Computational Linguistics, June 2009
11. Genzel, D.: Automatically learning source-side reordering rules for large scale machine translation. In: Proceedings of the 23rd International Conference on Computational Linguistics. COLING 2010, pp. 376–384. Association for Computational Linguistics, Stroudsburg (2010)
12. Lerner, U., Petrov, S.: Source-side classifier preordering for machine translation. In: EMNLP, pp. 513–523 (2013)
13. Papineni, Kishore, S.R.T.W., Zhu, W.: Bleu: a method for automatic evaluation of machine translation. In: ACL (2002)
14. Koehn, P., et al.: Moses: open source toolkit for statistical machine translation. In: Proceedings of ACL, Demonstration Session (2007)
15. Yang, N., Li, M., Zhang, D., Yu, N.: A ranking-based approach to word reordering for statistical machine translation. In: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1, pp. 912–920. Association for Computational Linguistics (2012)
16. Habash, N.: Syntactic preprocessing for statistical machine translation. In: Proceedings of the 11th MT Summit (2007)

17. Bach, N., Gao, Q., Vogel, S.: Source-side dependency tree reordering models with subtree movements and constraints. In: Proceedings of the Twelfth Machine Translation Summit (MTSummit-XII), Ottawa, Canada, International Association for Machine Translation, August 2009
18. Cai, J., Utiyama, M., Sumita, E., Zhang, Y.: Dependency-based pre-ordering for Chinese-English machine translation. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (2014)
19. Goto, I., Utiyama, M., Sumita, E., Kurohashi, S.: Preordering using a target-language parser via cross-language syntactic projection for statistical machine translation. ACM Trans. Asian Low-Resource Lang. Inf. Process. **14**(3), 13 (2015)
20. Daiber, J., Stanojevic, M., Aziz, W., Sima'an, K.: Examining the relationship between preordering and word order freedom in machine translation. In: Proceedings of the First Conference on Machine Translation (WMT 2016). Association for Computational Linguistics, Berlin, August 2016
21. Ding, C., Sakanushi, K., Touji, H., Yamamoto, M.: Inter-, intra-, and extra-chunk preordering for statistical Japanese-to-English machine translation. ACM Trans. Asian Low-Resour. Lang. Inf. Process. **15**(3), 20:1–20:28 (2016)
22. Hadiwinoto, C., Liu, Y., Ng, H.T.: To swap or not to swap? Exploiting dependency word pairs for reordering in statistical machine translation. In: Thirtieth AAAI Conference on Artificial Intelligence (2016)
23. Bansal, M., Gimpel, K., Livescu, K.: Tailoring continuous word representations for dependency parsing. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pp. 809–815, June 2014
24. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. In: International Conference on Learning Representations (ICLR) Workshop (2013)
25. Cer, D., de Marneffe, M.C., Jurafsky, D., Manning, C.D.: Parsing to Stanford dependencies: trade-offs between speed and accuracy. In: 7th International Conference on Language Resources and Evaluation (LREC 2010) (2010)
26. : Tensorflow: Large-scale machine learning on heterogeneous systems (2015). Software available from tensorflow.org
27. Luong, M., Brevdo, E., Zhao, R.: Neural machine translation (Seq2Seq) tutorial (2017). https://github.com/tensorflow/nmt
28. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. arXiv preprint arXiv:1607.04606 (2016)
29. Nguyen, T.P., Shimazu, A., Ho, T.B., Nguyen, M.L., Nguyen, V.V.: A tree-to-string phrase-based model for statistical machine translation. In: Proceedings of the Twelfth Conference on Computational Natural Language Learning (CoNLL 2008), Manchester, England, Coling 2008 Organizing Committee, August 2008, pp. 143–150 (2008)
30. Stolcke, A.: Srilm - an extensible language modeling toolkit. In: Proceedings of International Conference on Spoken Language Processing, vol. 29, pp. 901–904 (2002)
31. Och, F.J., Ney, H.: A systematic comparison of various statistical alignment models. Comput. Linguist. **29**(1), 19–51 (2003)
32. Tran, V.H., Vu, H.T., Nguyen, V.V., Nguyen, M.L.: A classifier-based preordering approach for English-Vietnamese statistical machine translation. 17th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing 2016) (2016)

The Utility of Hierarchical Phrase-Based Model Machine Translation for Low Resource Languages

S. Yashothara(✉) and R. T. Uthayasanker(✉)

Department of Computer Science and Engineering,
University of Moratuwa, Moratuwa, Sri Lanka
yashoshan@gmail.com, rtuthaya@cse.mrt.ac.lk

Abstract. The paper uses a hierarchical phrase-based model to develop Statistical Machine Translation (SMT) Systems for four low resourced South Asian languages. South Asian languages predominantly use traditional statistical and neural machine approaches to translate into another language (mainly English). However, translation accuracy is not much higher as South Asian languages lack in necessary natural language resources and tools; hence classified as low resourced languages. Any SMT system needs large parallel corpora for actual performance. So, the non-availability of corpora constraints the success in machine translation of those languages. Another reason for poor translation quality is grammatical differences between South Asian languages and English: morphological richness and different sentence structure. But traditional SMT systems use the default distortion reordering model to reorder the sentences independent of their context. To overcome this problem, hierarchical phrase model translation, which uses grammar rules formed by the Synchronous Context-Free Grammar, is proposed. This paper considers English to Tamil, Tamil to English, Malayalam to English, English to Malayalam, Tamil to Sinhala and Sinhala to Tamil translations. In the end, we evaluate the system using BLEU as the evaluation metric. The hierarchical phrase-based model shows better results compared to the traditional approach between Tamil-English and Malayalam-English pairs. For Sinhala to Tamil, it achieves 11.18 and 10.73 for vice-versa.

Keywords: Hierarchical phrase-based model · Statistical machine translation · Parallel corpus · Natural language processing

1 Introduction

With the rebellion of the internet, people are becoming more global. However, communication between the people is still challenging as they speak different languages. Though English is widely accepted as an official language in many multilingual South Asian countries like Sri Lanka and India, we can not assure that everyone knows it. Therefore, translation plays a significant role. Currently, we use traditional statistical and neural machine approaches for translation. But, most of these South Asian languages lack necessary natural language resources and tools.

Translation between the South Asian languages and English is still challenging due to the different nature of these languages than English. South Asian languages are morphologically rich and commonly use unique sentence structures. The structure of a sentence is Subject-Verb-Object in English while Subject-Object-Verb in most of the South Asian languages. The difference between these languages' colloquial and formal forms is also much more significant compared to English.

As South Asian languages are low resourced and have unique sentence order, it is difficult to get a good order of sentences (because of sub-phrases) when using traditional Statistical Machine translation (SMT). SMT uses a distortion reordering model to reorder the sentences, which is independent of their context. Also, learning phrases longer than three words barely improve the translation because such phrases are infrequent in the corpora due to data sparsity. Data sparsity is very likely that we will not see all of the words at training time but having similar text.

To overcome the above issues, we adopted hierarchical phrased-based machine translation [1, 2]; one of the current leading and promising SMT approaches. Hierarchical phrased-based translation extends the phrase-based translation through modelling as Synchronous Context-Free Grammar (SCFG) [3]. The hierarchical model brings sub-phrases into existence to remove the problems associated with phrase-based translation. Although the phrase-based reordering model captures global reordering, the reordering does not explicitly introduce the model to restrict word order within sub phrases. The hierarchical phrase-based model deals with reordering sub-phrases using nonterminal symbols and lexicalized reordering models [3].

Hierarchical phrase-based SMT [3] combines the strength of a rule-based and a phrase-based machine translation system. It constructs trees by automatically extracting a SCFG from the training corpus. The basic unit of hierarchical phrased-based SMT is hierarchical rules that are extracted by Context-Free Grammar [5]. So, hierarchical rules have the strength of learning sentence reordering without a separate reordering model.

A paramount concern with hierarchical phrase-based translation is the training model size, which is usually several times larger than the trained phrase-based counterpart from the same dataset. But, this heads to over generation search errors and a slow decoder [4].

The key focus of the research is the utility of the hierarchical phrase-based model in the translation of South Asian languages, inspired by Chiang's contribtuion [1]. Notably, this research focuses on Tamil to English, English to Tamil, Malayalam to English, English to Malayalam, Tamil to Sinhala and Tamil to Sinhala within all the South Asian languages.

We conducted experiments with hierarchical phrase-based translation using the Moses tool and compared it with traditional phrase-based models using the same corpora. We have selected the Tamil-Sinhala pair of languages to check the performance of the hierarchical model in the translation between two languages with the same sentence structure.

2 Literature Review

This section reviews the literature in two viewpoints: utility of the hierarchical phrased-based model in other language pairs and existing machine translation systems for Tamil, English, Malayalam and Sinhala languages.

2.1 Utility of the Hierarchical Phrased-Based Model

Chiang [3] firstly proposed a hierarchical phrase-based model for SMT. Their experiments were on Mandarin-to-English translation and claimed that the hierarchical phrase-based SMT system achieves an absolute improvement of 0.02 over the traditional SMT (7.5% relative) without any additional training data [3]. Mohaghegh and Sarrafzadeh [6] adopted this method for the translation between English and Persian languages. They have compared Moses tool kit and Joshua tool kit by dividing corpus into sections of 20 K, 30 K, 40 K, and 50 K sentences. The best result claimed in the paper is 4.5269 NIST and 0.3708 BLEU using the Joshua based system trained on 50 K corpus [6].

One of the early works on hierarchical phrase-based models for south Asian languages was proposed by Jawaid et al. [7], who examined English and Urdu. They experimented using the Moses SMT system and presented an Urdu aware approach based on reordering phrases in syntactic parse tree of the source English sentence [7]. Khan et al. [8] focused on English to Urdu hierarchical phrase-based SMT. Six thousand five hundred ninety-six sentences parallel corpus is used for training. The authors used the k-fold ($k = 5$) cross-validation method for sampling the corpus. The highest percentage of results is 29% in the experiment [8].

2.2 Existing Machine Translation Systems for Tamil, English, Malayalam and Sinhala Languages

There are many translation approaches for Tamil-English, Malayalam-English and Tamil-Sinhala language pairs. The following subsection provides the significant efforts of machine translation for these languages.

Ulrich Germann [9] conveyed his experience by building a SMT system for translation between Tamil and English from scratch, including creating a small parallel Tamil-English corpus. Following this research, several other research studies [10, 11] use traditional SMT. Loganathan [12] developed a SMT system by integrating morphological information. He separated the morphological suffixes to improve the quality of the traditional phrase-based model [12]. Anandkumar et al. [13] adopted a factored SMT system to handle the morphologically fluent Tamil sentences. They applied the manually created reordering rules to the syntactic trees for rearranging the phrases in English. It improves the performance in local distance sentences and already available sentences in the training corpora [13]. But long-distance reordering and new sentence reordering are not handled in these approaches.

The first effort of Malayalam to English translation is a rule-based system [11]. But, rule-based systems' development requires more cost and time for linguistic rules; and it sometimes fails to find good translation due to search errors during the decoding process. Sebastian et al. [15] proposed a SMT approach by adding some pre-processing and post-processing steps. The alignment model is improved by adding parts of speech information into the bilingual corpus and removing the improper alignments from the sentence pairs. Corpus is pre-processed by suffix and stop word elimination techniques. They have used order conversion rules to resolve the structural difference between English and Malayalam languages [15]. But, adding rules to translation also faces problems such as high cost in formulating rules and conflicts when the numbers of rules increase.

There is very minimal research for the local languages of Sri Lanka (Sinhala -Tamil). As the first attempt, Ruvan Weerasinghe [16] proposed a basic SMT approach to Sinhala and Tamil languages. After testing with multiple translation models, they have achieved a best BLEU score of 0.1362 [16]. Following that work, S. Sripirakas et al. [17] proposed a translation system implemented on parallel corpora from parliament order papers. They demonstrate only the preliminary system, which runs both directions of Tamil and Sinhala languages [17]. There are some similar approaches [18–21] by using SMT. But there have been no efforts to translate between Tamil and Sinhala languages using hierarchical phrase-based SMT machine translation. But, traditional SMT mostly fails to produce quality output for long sentences.

By looking at the work above, it is clear that there is no single proposed work in hierarchical phrase-based models for Tamil, Malayalam and Sinhala.

3 Experiment

This section discusses the training, tuning and testing of different model components. The experiment was conducted on Ubuntu 16.00 running on an Intel Core i5 machine with 2 GB of RAM and 500 GB of Hard disk space between Tamil-English, Malayalam-English and Tamil-Sinhala.

3.1 Dataset

We used IIIT-Hyderabad (International Institute of Information Technology) parallel corpus for Tamil-English and Malayalam-English languages. The corpora contain datasets of eleven languages. The size of each corpus is about 3 million words. Each corpus's text is categorized under aesthetics, mass media, social science, natural science, commerce, and translated materials. The corpora were prepared by several organizations under the funding from MoIT (Ministry of Information Technology formerly Department of Electronics), Government of India. Its bilingual resources consist of roughly about 50,000 sentences for all the available languages [22]. The corpora are already sentence aligned. Here we clean the corpus for making it completely compatible.

The primary source of the parallel corpus of Sinhala-Tamil languages is government official documents. The documents collected from government institutions were hard copies, and some were of a single source. They are generally digitalized translated manually with the aid of human translators. Its bilingual resources consist of about 22,000 sentences for all the available languages. Table 1 shows the statistics of the parallel corpora.

Table 1. Complete statistics of the parallel corpus (in sentence)

	Tamil-English	Malayalam-English	Tamil-Sinhala
Training	48,000	48,000	20,000
Development	1,500	1,500	1,500
Testing	500	500	500

We use the target language corpus in the above parallel corpus to develop a language model for this study work.

3.2 Experimental Setup

We conduct the experiments to check the utility of the hierarchical phrase-based model in translation between morphologically rich languages (Tamil-Sinhala) and morphologically rich-poor languages (Tamil-English, Malayalam-English).

As the initial step of the experiments, we tokenized the obtained data using customized scripts. We removed the sentences with extreme length ratio differences using standard Moses [23] filtration. We used the script supplied with the Moses decoder [23] to lowercase the English language corpus. Then, we aligned the trained data using the provided word alignment tool. Lastly, we trained the system with Koehn's training scripts. In our work, additional switches like hierarchical and glue grammar were also used in training commands.

The default values were used for the other parameters, i.e. 3-g language model and maximum phrase length = 6. We used Giza+ + [25] for the word alignment with 'grow-diag-final-and' as the summarization heuristics and Lmplz [24] for the language modelling (3-g). We tuned the features with Minimum Error Rate Training (MERT) on 100 best translations [23] on a set of 1500 randomly selected sentences. Decoding was done using the state-of-the-art Moses using cube pruning techniques with a stack size of 5000 and a maximum phrase length of 5 [26]. The testing was carried out in the same way for all the language pairs with Moses decoder. To compare the results of hierarchical phrase-based SMT, we also built a traditional SMT approach for the same data set.

The output of the system was evaluated using Bilingual Evaluation Understudy (BLEU) [27]. The system was evaluated on 500 randomly selected sentences/phrases. The letter headers and footers were added as comma-separated phrases for testing to ensure that the score of a single sentence no longer depends on a single or minimal amount of words.

4 Results

The evaluation scores of the three languages pairs and the sample translations from the developed hierarchical phrase-based SMT are described in this section. In each language pair, we trained the SMT with and without the hierarchical phrase-based model. We evaluated its translation quality by measuring the BLEU score of the translation of the test data set. Even though these language resources are sparse, we achieved a better BLEU score for the entire language pairs. We tabulate the scores of six different experimental setups in Table 2. A comparison of the developed hierarchical phrase-based translation system with the traditional phrase-based system was also carried out for the same dataset.

Table 2 shows that the hierarchical phrase-based SMT system got better BLEU scores compared to the traditional phrase-based model approach for Tamil to English, English to Tamil, Malayalam to English and English to Malayalam. Though those differences are minor since the dataset size is small, the percentage of the difference is high. These results show that the usefulness of a hierarchical phrase-based model is significant when there is a difference in sentence structure between the languages. Nevertheless, for the translation of Tamil to Sinhala and Sinhala to Tamil, it could be noticed from Table 2 that the traditional phrase-based model system got better BLEU scores compared to the hierarchical phrase-based model approach. The main reason behind this is that both Tamil and Sinhala languages share the same sentence structure and are morphologically rich.

Further, the Tamil-Sinhala corpus is the smallest among the three, which causes sparseness in training data. Hierarchical phrase-based model is sensitive to sparse data, which could have further reduced the translation quality. These observations show that the hierarchical phrase-based model is most useful in language pairs varied by sentence structure but would affect the quality of the translation if the languages share the same sentence structure.

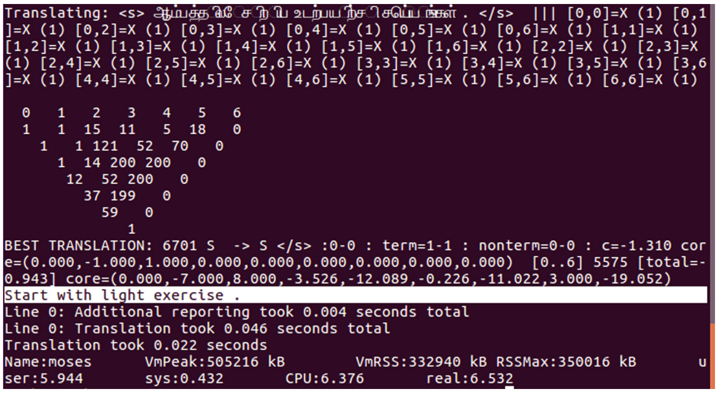
Table 2. Comparison of BLEU evaluation score between hierarchical phrase-based SMT and traditional SMT models

	BLEU Score		Differentiation (%)
	Traditional SMT	Hierarchical Model	
Tamil to English	3.16	3.42	8.23
English to Tamil	1.17	1.73	47.863
Malayalam to English	4.22	4.40	4.26
English to Malayalam	2.88	3.310	14.93
Tamil to Sinhala	*14.88	11.18	(−20.16)
Sinhala to Tamil	*13.61	10.73	(−21.17)

The results also show that the BLEU score improvement is greater from English to Tamil or Malayalam than the other direction due to the morphological richness of these languages. In these cases, the hierarchical phrase-based model leverages the morphological divergence between these languages in its favour. We noted that translating from morphologically rich (Tamil, Malayalam) to morphologically poor (English) gives a better BLEU score in both systems (traditional and hierarchical phrase-based SMT). Even Sinhala is morphologically rich, the 'Tamil to Sinhala' translation shows higher results, as Tamil is morphologically richer than Sinhala. The observations show that the translations from morphologically rich to morphologically poor give better results than in another direction.

Also, English to Tamil got the highest increase in BLEU score (47%), and Sinhala to Tamil got the highest reduction in BLEU score percentage (21%). Results show that the translations from morphologically poor (English) to morphologically rich (Tamil, Malayalam) improve using the hierarchical phrase-based model. So, the usefulness of the hierarchical phrase-based model is significant when there is a divergence of morphology and sentence structure.

Figure 1 shows how the decoder translations of the test dataset using the chart decoder for the hierarchical phrase-based model. For the input Tamil sentence "ஆரம்பத்திலே சிறிய உடற்பயிற்சி செய்யுங்கள்.", the sentence is translated as "Start with light exercise".



```
Translating: <s> ஆரம்பத்திலே சிறிய உடற்பயிற்சி செய்யுங்கள் . </s>  ||| [0,0]=X (1) [0,1
]=X (1) [0,2]=X (1) [0,3]=X (1) [0,4]=X (1) [0,5]=X (1) [0,6]=X (1) [1,1]=X (1)
[1,2]=X (1) [1,3]=X (1) [1,4]=X (1) [1,5]=X (1) [1,6]=X (1) [2,2]=X (1) [2,3]=X
(1) [2,4]=X (1) [2,5]=X (1) [2,6]=X (1) [3,3]=X (1) [3,4]=X (1) [3,5]=X (1) [3,6
]=X (1) [4,4]=X (1) [4,5]=X (1) [4,6]=X (1) [5,5]=X (1) [5,6]=X (1) [6,6]=X (1)

  0   1   2   3   4   5   6
  1   1  15  11   5  18   0
    1   1 121  52  70   0
      1  14 200 200   0
       12  52 200   0
         37 199   0
           59   0
              1
BEST TRANSLATION: 6701 S  -> S </s> :0-0 : term=1-1 : nonterm=0-0 : c=-1.310 cor
e=(0.000,-1.000,1.000,0.000,0.000,0.000,0.000,0.000,0.000)  [0..6] 5575 [total=-
0.943] core=(0.000,-7.000,8.000,-3.526,-12.089,-0.226,-11.022,3.000,-19.052)
Start with light exercise .
Line 0: Additional reporting took 0.004 seconds total
Line 0: Translation took 0.046 seconds total
Translation took 0.022 seconds
Name:moses      VmPeak:505216 kB        VmRSS:332940 kB RSSMax:350016 kB       u
ser:5.944       sys:0.432       CPU:6.376       real:6.532
```

Fig. 1. Tamil to English translation of hierarchical phrase-based model decoder.

Some examples of translation generated by the translation system are provided in Table 3. Some of the examples are not perfect translations. The reasons for the non-perfect translations are: morphologically richness, noise in the training data, out of vocabulary, misordering of words, wrong alignment of phrases, inappropriate translation according to the context and more complex sparse data (due to various sources for the data). However, the hierarchical model improves the quality of reordering the phrases.

Table 3. Example translations of hierarchical phrase-based SMT

	Input	Output
Tamil to English	சைட்டிகா மற்றும் சிலிப்புடிஸ்க் நோயாளி இந்த பயிற்சி செய்யாதீர்கள்	The patients of sciatica and slip disk should avoid its practice
	பூங்காவிற்கு பைக் எடுத்துச்செல்ல அனுமதியில்லை.	The a rate of a take her அனுமதியில்லை
English to Tamil	Drink plenty of water	நன்றாக தண்ணீர் குடியுங்கள்
	Chew the sugar -free chewing gum	சர்க்கரை இல்லாத சூயிங்கம் மெல்லவேண்டும்
Malayalam to English	പതിവായിട്ട് ദന്തനിരീക്ഷണം ചെയ്യണം .	Get the teeth checked -up regularly
	നെഓറാ താഴ്വര ദേശീയ ഉദ്യാനത്തില് അനേകം ഓര്ക്കിഡുകളുണ്ട് .	Neora Valley National Park is Approximately 150 species of orchids are found
English to Malayalam	New Digha is the new tourist spot of Digha	ചിന്ഢിയിലെ കമരൂനാഗ് . ഗിവഗുഫ തുടങ്ങിയവ കാണേണ്ടതാണ് .
	There is ` Forest Hut ' in Sanarali 3 kms ahead	3 കി.മീ. സനാരലിയില് ഫാം റെസ്റ്റ് ഹട്ട് ഉണ്ട് .
Tamil to Sinhala	கலந்துரையாடல் நிகழ்வு செயலாளரின் தலைமையில் இடம்பெற்றது	සංවාද සිද්ධිය ලේකම්ගේ ප්‍රධානත්වයෙන් පැවැත්විණ .
	பணிக்கான கணக்குப் பிரிவு	කාර්යය ගිණුම් අංශය
Sinhala to Tamil	ජල පොම්ප අළුත්වැඩියා අංශය	நீர் திருத்துதல் பிரிவு
	ජල සැපයුම් අංශය	நீர் வழங்கல் பிரிவு

5 Conclusion

We explored one of the significant but relatively less addressed research problems. Our research is the first to develop a hierarchical phrase-based SMT for some South Asian languages. In this work hierarchical phrase-based model was applied in the translations of Tamil to English, English to Tamil, Malayalam to English, English to Malayalam, Tamil to Sinhala and Sinhala to Tamil. This study compared traditional SMT and hierarchical phrase-based SMT for the translation between South Asian and English languages. We observed that hierarchical phrase-based SMT outperforms the traditional SMT for translating morphologically rich and poor (for the same dataset). The hierarchical phrase-based model helps to improve the translation quality between languages that vary by sentence structure. However, in the Sinhala-Tamil translation, the traditional approach performs better than the hierarchical phrase model. The reasons are the high sensitivity of the hierarchical phrase-based to sparse data and no built-in parser for other languages except English in the Moses tool. One of the limitations of our research is the lack of freely available linguistic resources and the shortage of well-developed and widely used open-source frameworks for the South Asian languages. We plan to analyze problems with existing datasets, integrate morphology, and enhance output quality in our future work.

Acknowledgement. We are grateful to Treesa Anjaly Cyriac, who supported in providing Malayalam resources. We are thankful to all our colleagues who supported the Linguists who helped align POS tagsets and us. The authors would also like to thank the LK Domain registry for partially funding this publication.

References

1. Chiang, D.: A hierarchical phrase-based model for statistical machine translation. In: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics. Association for Computational Linguistics (2007)
2. Watanabe, T., Tsukada, H., Isozaki, H.: Left-to-right target generation for hierarchical phrase-based translation. 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 (2006)
3. Chiang, D.: A hierarchical phrase-based model for statistical machine translation. In: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics. Association for Computational Linguistics (2005)
4. Gispert, A.: Hierarchical phrase-based translation with weighted finite-state transducers and shallow-n grammars. Comput. Linguist. **36**(3), 505–533 (2010)
5. Koehn, P.: Edinburgh system description for the 2005 IWSLT speech translation evaluation. In: International Workshop on Spoken Language Translation (IWSLT) 2005 (2005)
6. Mohaghegh, M., Sarrafzadeh, A.: A hierarchical phrase-based model for English-persian statistical machine translation. In: 2012 International Conference on Innovations in Information Technology (IIT) (2012)
7. Jawaid, B., Zeman, D.: Word-order issues in english-to-urdu statistical machine translation. The Prague Bulletin of Mathematical Linguistics **95**, 87–106 (2011)
8. Khan, N.: English to urdu hierarchical phrase-based statistical machine translation. In: Proceedings of the 4th Workshop on South and Southeast Asian Natural Language Processing (2013)
9. Germann, U.: Building a statistical machine translation system from scratch: how much bang for the buck can we expect? In: Proceedings of the Workshop on Data-Driven Methods in Machine Translation, pp. 1–8. ACL, Morristown, NJ, USA (2001)
10. Vasu Renganathan.: An interactive approach to development of English to Tamil machine translation system on the web. In: Proceedings of INFITT-2002 (2002)
11. AUKBC Research Centre
12. Loganathan, R.: English-Tamil Machine Translation System. Master of Science by Research Thesis, Amrita Vishwa Vidyapeetham, Coimbatore (2010)
13. Kumar, M.A.: Factored statistical machine translation system for English to Tamil language. Pertanika J. Soc. Sci. Humanit. **22**(4) (2014)
14. Unnikrishnan, P., Antony, P.J., Soman, K.P.: A novel approach for English to south dravidian language (2011)

15. Sebastian, M.P., Sheena Kurian, K., Kumar, G.S.: A framework of statistical machine translator from English to malayalam. In: Proceedings of Fourth International Conference on Information Processing, Bangalore, India (2010)
16. Weerasinghe, R.: A statistical machine translation approach to sinhala-tamil language translation. Towards an ICT Enabled Society, p. 136 (2003)
17. Sripirakas, S., Weerasinghe, A., Herath, D.L.: Statistical machine translation of systems for Sinhala-Tamil. In: International Conference on Advances in ICT for Emerging Regions (ICTer), 2010 (2010)
18. Sripirakas, S., Weerasinghe, A.R., Herath, D.L.: Statistical machine translation of systems for Sinhala-Tamil. In: International Conference on Advances in ICT for Emerging Regions (ICTer), 2010, pp. 62–68. IEEE (2010)
19. Jeyakaran, M., Weerasinghe, R.: A novel kernel regression based machine translation system for Sinhala-Tamil translation. In: Proceedings of the 4th Annual UCSC Research Symposium (2011)
20. Pushpananda, R., Weerasinghe, R., Niranjan, M.: Sinhala-Tamil machine translation: towards better translation quality. In: Australasian Language Technology Association Workshop 2014, Melbourne (2014)
21. Rajpirathap, S., Sheeyam, S., Umasuthan, K.: Real-time direct translation system for Sinhala and Tamil languages. In: 2015 Federated Conference on Computer Science and Information Systems (FedCSIS) (2015)
22. http://ltrc.iiit.ac.in/corpus/corpus.html
23. Koehn, P., et al.: Moses: open source toolkit for statistical machine translation (2007)
24. Heafield, K.: KenLM: faster and smaller language model queries. In: Proceedings of the Sixth Workshop on Statistical Machine Translation (2011)
25. Och, F.: Minimum error rate training in statistical machine translation. In: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics (2003)
26. Huang, L., Chiang, D.: Forest rescoring: faster decoding with integrated language models. In: Proceedings of the Annual Meeting-Association For Computational Linguistics (2007)
27. 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 (2002)
28. Nagata, M., et al.: A clustered global phrase reordering model for statistical machine translation. 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 (2006)
29. Pushpananda, R., Weerasinghe, R., Niranjan, M.: Statistical machine translation from and into morphologically rich and low resourced languages. In: Gelbukh, A. (ed.) CICLing 2015. LNCS, vol. 9041, pp. 545–556. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-18111-0_41

Automatic Method to Build a Dictionary for Class-Based Translation Systems

Kohichi Takai[1](✉), Gen Hattori[1], Keiji Yasuda[1], Panikos Heracleous[1], Akio Ishikawa[1], Kazunori Matsumoto[1], and Fumiaki Sugaya[1,2]

[1] KDDI Research, Inc., Garden Air Tower, 3-10-10, Iidabashi, Chiyoda-ku, Tokyo 102-8460, Japan
{ko-takai,ge-hattori,ke-yasuda,pa-heracleous, ao-ishikawa,matsu}@kddi-research.jp

[2] MINDWORD Inc., 7-19-11 Nishishinjuku, Shinjuku-ku, Tokyo 160-0023, Japan
fsugaya@mindword.jp

Abstract. Mis-translation or dropping of proper nouns reduces the quality of machine translation or speech translation output. In this paper, we propose a method to build a proper noun dictionary for the systems which use class-based language models. The method consists of two parts: training data building part and word classifier training part. The first part uses bilingual corpus which contain proper nouns. For each proper noun, the first part finds out the class which gives the highest sentence-level automatic evaluation score. The second part trains CNN-based word class classifier by using the training data yielded by the first step. The training data consists of source language sentences with proper nouns and the proper nouns' classes which give the highest scores. The CNN is trained to predict the proper noun class given the source side sentence. Although, the proposed method does not require the manually annotated training data at all, the experimental results on a statistical machine translation system show that the dictionary made by the proposed method achieves comparable performance to the manually annotated dictionary.

Keywords: Machine translation · Speech translation · Proper noun dictionary · Named entity

1 Introduction

As a result of drastic advances in technical innovations of speech processing and natural language processing, a speech-to-speech translation system is becoming a realistic tool for travelers. Especially for the travel domain, the coverage of proper nouns for tourist spots, landmarks, restaurants, and accommodations highly influences system performance.

Okuma et al. [1] proposed a class based method to install hand-crafted bilingual dictionaries into a Statistical Machine Translation (SMT) framework to improve the proper noun coverage of Machine Translation (MT). However, there are remaining problems for practical usage. It is the cost of building the dictionary. Since the number of proper noun such as restaurant names or product names, is increasing daily, the cost to manually maintain a dictionary is very high. Especially for the speech translation system using a class-based language model, there is an additional cost to annotate word class for each new word. In this paper, we propose a method to automatically estimate word class.

Section 2 describes related work in this research. Section 3 explains the proposed method. Section 4 shows the experimental results using the SMT-based system. Section 5 concludes the paper and presents some directions for future work.

2 Related Work

Class-based language model had been proposed in the automatic speech recognition research field to solve the problem of data sparseness. And, the idea also has been applied to SMT framework [1]. To apply class based idea to SMT framework, we have to maintain dictionary which includes translation pairs of words and their word classes. To automatically acquire translation pairs, several methods have been proposed, such as bilingual dictionary [2] and transliteration approach [3–5].

In this paper, we only focus on the word class annotation part. There are also many researches dealing with automatic annotation of word category, in named entity recognition and machine translation research field [6,7]. These methods require manually annotated training data to train a classifier. The proposed method in this paper also uses supervised training approach. However, the proposed method automatically builds a training data by using bilingual corpus and class-based machine translation system which is the target system to use the dictionary. Although the bilingual corpus is required to build the training data, only monolingual corpus is required to actual word class classification.

3 Proposed Method

As shown in Fig. 1, the proposed method consists of training data building part and word classifier training part. Details of these two parts are explained in this section.

3.1 Training Data Building Without Human Annotation

Figure 2 shows the flow of training data building part. Broken lines in the figure indicate data flow. The first part uses bilingual corpus which contains proper noun. For each bilingual sentence pairs with proper nouns, the proposed method yields multiple machine translation results by the following steps.

Step1 Translation pairs of proper noun are extracted from bilingual sentence pairs from the corpus.

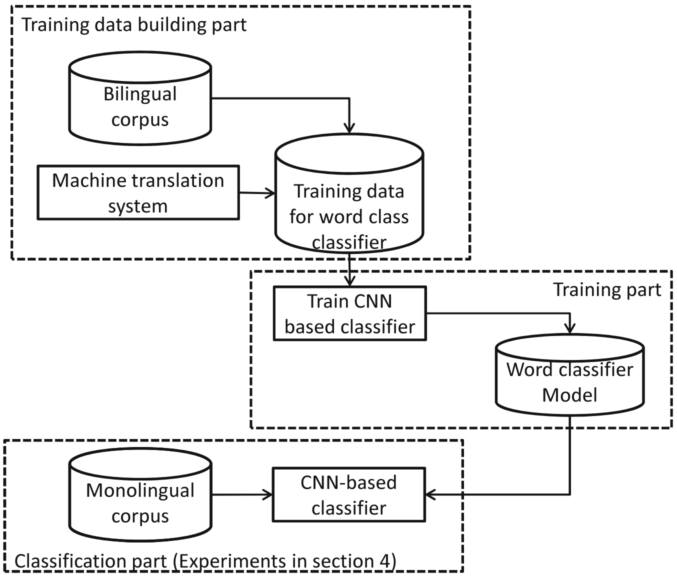

Training data building part
Bilingual corpus
Machine translation system
Training data for word class classifier
Training part
Train CNN based classifier
Word classifier Model
Monolingual corpus
CNN-based classifier
Classification part (Experiments in section 4)

Fig. 1. Framework for the proposed method.

Step2 The extracted proper noun pairs are registered to the dictionary as one of word classes.
Step3 Translate the source side sentence containing the target proper noun using MT with the dictionary made in step2.
Step4 The registered proper noun entry is removed from the dictionary.
Step5 Step 3 to 4 are repeated until all classes are registered to the dictionary.

By using the multiple translations and reference sentence which is the target language side of the sentence translation pair, the most suitable word class ($\hat{c}$) for the target proper noun is chosen using the following formula.

$$\hat{c} = \mathrm{argmax}_{c \in C}\ S_{RIBES}(T_{REF}, T_{MT}^{c}) \tag{1}$$

where C, T_{REF} and T_{MT}^{c} are the set of word class, reference translation which is the target side sentence from bilingual corpus, and a translation result by MT whose dictionary has entry of target proper noun as word class c, respectively. And S_{RIBES} is the RIBES [8] score between T_{REF} and T_{MT}^{c}, calculated using the following formula.

$$S_{RIBES}(T_{REF}, T_{MT}^{c}) = R_{cor}(T_{REF}, T_{MT}^{c}) \times (l_{com}/l_{MT})^{\alpha} \tag{2}$$

where l_{MT}, l_{com} and R_{cor} are the total words number in T_{MT}^{c}, the number of common words between T_{REF} and T_{MT}^{c} and the rank correlation coefficient between the common words, respectively. $\alpha(0 \leqq \alpha \leqq 1)$ is the hyper parameter for penalty.

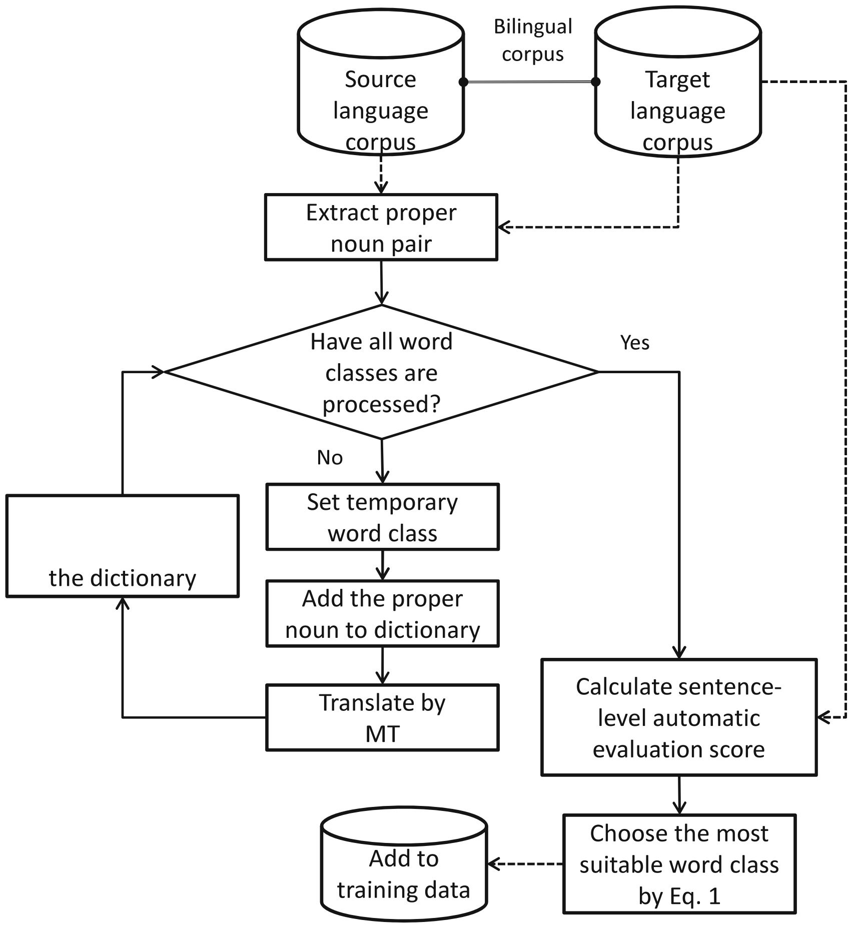

Bilingual corpus
Source language corpus
Target language corpus
Extract proper noun pair
Have all word classes are processed?
Yes
No
Set temporary word class
Add the proper noun to dictionary
Translate by MT
the dictionary
Calculate sentence-level automatic evaluation score
Choose the most suitable word class by Eq. 1
Add to training data

Fig. 2. Flow of the traning data building part.

In the word classifier training part, pairs of source side sentence and $\hat{c}$ which is teacher signal are used for training.

3.2 Word Classifier Training Part

This subsection explains the method to train word classifier using the training set build by the previous subsection.

For proper noun classification, we train a classifier which is configured as a Convolutional Neural Network (CNN). CNN showed superior performance in the research fields of image processing and speech recognition [9,10]. Currently, CNN has also outperformed the Natural Language Processing (NLP) task such

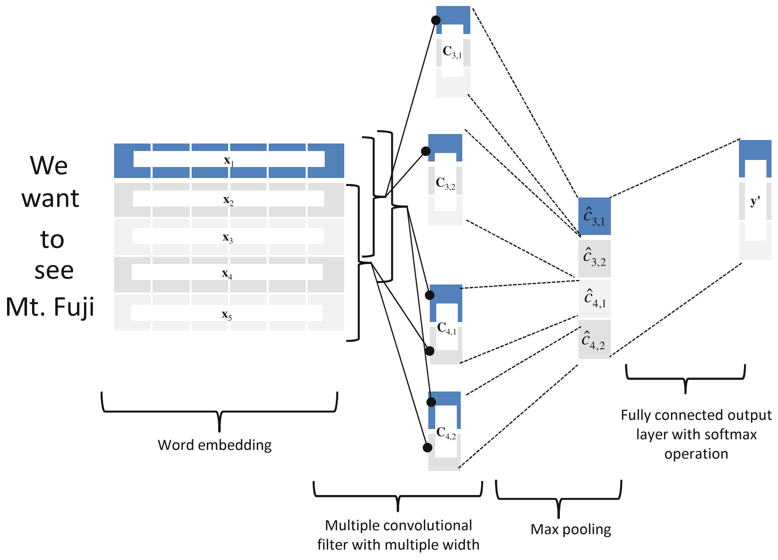


Fig. 3. Convolutional neural network for word classification.

as text classification [11,12] by incorporating word-embedding [13] in the input layer.

The network configuration for our word classification is shown in Fig. 3[1]. Here, $\mathbf{x}_i (\in \mathbb{R}^k)$ is the word-embedding vector of i-th word in a given sentence. By concatenating word-embedding vectors, a sentence whose length is n is expressed as follows:

$$\mathbf{x}_{1:n} = \mathbf{x}_1 \cdots \oplus \mathbf{x}_i \cdots \oplus \mathbf{x}_n \tag{3}$$

The convolutional layer maps the n-gram features whose length (or filter window size)is h to the j-th feature map by using the following formula:

$$c_{h,j,i} = \tanh(\mathbf{w}_{h,j} \cdot \mathbf{x}_{i:i+h-1} + b_{h,j}) \tag{4}$$

where $w_{h,j}$ and $b_{h,j}$ are weights for filtering and bias terms, respectively. For each n-gram length and feature map number, concatenate the results of Eq. 4 as follows

$$\mathbf{c}_{h,j} = [c_{h,j,1}, c_{h,j,2}, \cdots, c_{h,j,n-h+1}] \tag{5}$$

The max pooling layer chooses the element that has the largest value from all elements in $\mathbf{c}_{h,j}$ as follows

$$\hat{c}_{h,j} = \max_{i=1}^{n-h+1} \mathbf{c}_{h,j} \tag{6}$$

[1] The actual hyper parameters' setting is different from the example shown in the figure. Detail setting will be explained in Sect. 4.

Table 1. Detaisl of word class in the data set.

Word class	% in the data set
Accommodation	10.33
Attraction	4.90
Building	8.44
Country name	12.15
Foreign First Name	5.57
Foreign Last Name	3.73
Food	7.84
Japanese First Name	4.35
Japanese Last Name	4.17
Land Mark	11.73
Organization	9.29
Shop	4.75
Souvenir	6.54
Others	6.20
Total	100

Table 2. Statistic of training corpora used for the experiments.

Corpus type	# of words	Lexicon size
BTEC (Japanese side)	83,942	9,266
Wikipedia corpus	10,363,151	116,556

Fully connected output layer takes the softmax operation to yield the probability distribution of the word class ($\hat{\mathbf{y}} \in \mathbb{R}^{n_c}$) as follows

$$\hat{\mathbf{y}} = \frac{\exp(z_q)}{\sum_{p=1}^{n_c} \exp(z_p)}, q = 1, \cdots, n_c \tag{7}$$

where n_c is the total number of word classes. And, $\mathbf{z}$ ($\in \mathbb{R}^{n_c}$) are the raw output values from output layer.

4 Experiments

4.1 Evaluation Method

In the experiments, firstly, we build the training data and train the CNN-based classifier by the proposed method. Then, we build Japanese-English bilingual proper noun dictionary by using the classifier. To evaluate dictionary quality, we compare Japanese-to-English translation quality of SMT-based system with several conditions as follows.

Table 3. CNN parameter setting

Parameters	Setting
Maximum length of input sentence	150 words
Mini batch size	65
Dimension of word-embedding vector (k)	400
Filter window size (n-gram length)	3 to 5-gram
Number of filters for each window size	128
Drop out rate for fully connected layer	0.5
Optimizer	Adam optimizer
# of output units	14

Condition 1 Manual word class annotation by a human annotator.
Condition 2 Random annotation with uniform distribution.
Condition 3 Random annotation with prior distribution shown in Table 1.
Condition 4 Automatic word class annotation by the propose method.

We also use RIBES [8] for automatic evaluation of the translation quality. And, the evaluation experiments are carried out by 10-closs validation manner.

4.2 Experimental Settings

For the extraction of proper nouns, we use Japanese and English parts of Basic Travel Expression Corpus (BTEC) [14]. By using an in-house dictionary, we extract 5,471 sentence translation pairs containing proper noun. Here, we extract sentences which only contain one proper in each sentence. Table 1 shows the details of the word class specification and occurrence in the training set. There are 14 word classes which are related to travel domain. In the data set, word class of country name has the highest occurrence. Meanwhile, foreign last name has the lowest occurrence.

Beforehand of training CNN-based classifier, we trained the word-embedding matrix using Word2Vec [13] on the Wikipedia corpus. The word-embedding matrix is fixed through CNN training. The detail of these two corpora are shown in Table 2.

Table 3 shows the detailed parameter setting of the CNN-based classifier. As shown in the table, the CNN has 14 output units, and each of them outputs the probabilities of the word classes, which are shown in Table 1. As shown in Table 1, data size varies depending on the word class. To reduce the adverse effect of data imbalances, we sampled training data to be balanced while mini batch training.

4.3 Experimental Results

Fig. 4 shows the automatic evaluation results of translation by SMT with several kinds of class-based bilingual dictionaries. In the figure, the vertical axis

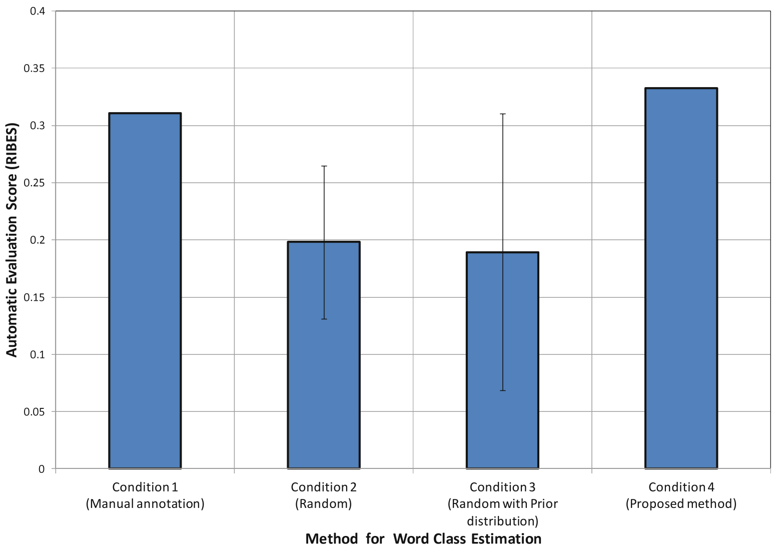


Fig. 4. Translation quality by SMT with several dictionaries.

represents automatic evaluation score calculated by RIBES. For all conditions, we use the same class-based SMT system proposed by [1] and same proper noun coverage. Difference is only the way to annotate word class for the proper noun dictionary. For condition 2 and 3, we carried out 10 times random annotations for each proper noun, then calculate an average score. The error bars in the figure show standard deviations over 10 times random trials. Meanwhile, for condition 4, we carried out 10-cross validation once on the data set.

As shown is the figure, the proposed method gives way better performance comparing to two random annotation. Additionally, the proposed method gives better performance than the manual annotation. The reason may be as follows. A human annotator decided word class from the proper nouns only. Meanwhile, the proposed method automatically annotates word class using the source side sentence including the proper noun, thus, the proposed method can use context information to annotate the proper noun. Such context information gives advantage for proper nouns which have multiple meanings.

5 Conclusions and Future Work

We proposed a method to build a dictionary for a class-based translation system.

We carried out experiments using BTEC corpus on SMT. Firstly, training set for CNN-based word classifier was automatically made. Secondly, the CNN-based word classifier was trained using the training set. Then, we automatically labeled word class for the proper nouns in the test set and added to the dictionary for SMT. Finally, we translate the test set by SMT using the dictionary to evaluate the dictionary quality by translation quality. According to the experimental results, the dictionary build by the proposed method have performance comparable to the manually annotated dictionary.

Since the proposed method does not require manual annotation, it enables quick development of class-based machine translation system.

As future work, we will increase the corpus size for improvement of classification accuracy. Although the experiments shown in the paper were carried out on the existing BTEC corpus, now we are carrying out speech translation field experiments to evaluate the effectiveness of the proposed method in the real field.

Acknowledgments. This research is supported by Japanese Ministry of Internal Affairs and Communications as a Global Communication Project.

References

1. Okuma, H., Yamamoto, H., Sumita, E.: Introducing a translation dictionary into phrase-based SMT. IEICE Trans. Inf. Syst. **91-D**, 2051–2057 (2008)
2. Tonoike, M., Kida, M., Takagi, T., Sasaki, Y., Utsuro, T., Sato, S.: Translation estimation for technical terms using corpus collected from the web. In: Proceedings of the Pacific Association for Computational Linguistics, pp. 325–331 (2005)
3. Al-Onaizan, Y., Knight, K.: Translating named entities using monolingual and bilingual resources. In: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL), pp. 400–408 (2002)
4. Sato, S.: Web-based transliteration of person names. In: Proceedings of IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, pp. 273–278 (2009)
5. Finch, A., Dixon, P., Sumita, E.: Integrating a joint source channel model into a phrase-based transliteration system. Proc. NEWS **2011**, 23–27 (2011)
6. 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 (2016)
7. Yasuda, K., Heracleous, P., Ishikawa, A., Hashimoto, M., Matsumoto, K., Sugaya, F.: Building a location dependent dictionary for speech translation systems. In: 18th International Conference on Computational Linguistics and Intelligent Text Processing (2017)
8. Isozaki, H., Hirao, T., Duh, K., Sudoh, K., Tsukada, H.: Automatic evaluation of translation quality for distant language pairs. In: Conference on Empirical Methods in Natural Language Processing, pp. 944–952 (2010)
9. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems 25, pp. 1097–1105. Curran Associates, Inc. (2012)
10. 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 Language Process. **22**, 1533–1545 (2014)
11. Kim, Y.: Convolutional neural networks for sentence classification. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, pp. 1746–1751 (2014)

12. Kalchbrenner, N., Grefenstette, E., Blunsom, P.: Convolutional neural networks for modeling sentences. In: Proceedings of the 52nd Annual Meeting for Computational Linguistics, pp. 655–665 (2014)
13. 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 26, pp 3111–3119. Curran Associates, Inc. (2013)
14. Kikui, G., Sumita, E., Takezawa, T., Yamamoto, S.: Creating corpora for speech-to-speech translation. In: 8th European Conference on Speech Communication and Technology (EUROSPEECH), pp. 381–382 (2003)

Addressing the Issue of Unavailability of Parallel Corpus Incorporating Monolingual Corpus on PBSMT System for English-Manipuri Translation

Amika Achom[1](✉), Partha Pakray[1], and Alexander Gelbukh[2]

[1] National Institute of Technology, Aizawl, Mizoram, India
achomamika01@gmail.com
[2] Centro de Investigación en Computación (CIC) of the Instituto Politécnico Nacional (IPN), Mexico, Mexico
gelbukh@cic.ipn.mx

Abstract. This research paper work establishes an important concept of improving Phrase based Statistical Machine Translation System incorporating monolingual corpus on the target side of the English to Manipuri translation language pair. However, there has been no work that focuses on translating one of the Indian Minority Tibeton-Burman Manipuri language pair. This Phrase based Statistical Machine Translation system has been developed using the Moses open-source toolkit and evaluated carefully using various automatic and human evaluation techniques. PBSMT achieves a BLEU Score of 10.15 as compared to the baseline PBSMT of BLEU Score 9.89 using the same training, tuning, and testing datasets. This research paper work addresses the issue of limited availability of parallel text corpora (English-Manipuri pair).

Keywords: Phrase based statistical machine translation · English to manipuri parallel corpora · Automatic-bilingual evaluation understudy score (BLEU) · Human evaluation-fluency · Adequacy · Overall rating

1 Introduction

Machine Translation (MT) is the automatic translation from one source language to any another target language. MT is one of the most dominant emerging fields of today's world. MT is a key to success to any new services. This has promoted the development of many new models and resulted in the development of an open source Statistical Machine Translation (SMT) system, Moses,[1] which is used in various institutes, research project and so on.

According to the latest Census Survey of India[2], it is reported that, 1.5 million people speak Manipuri language or Meiteilon. This arises the need of

[1] http://www.statmt.org/moses/?n=Development.GetStarted.
[2] https://en.wikipedia.org/wiki/Languages-of-India.

MT technology to translate any low-level Manipuri language. There are different approaches to MT. They are undermentioned:

Rule-based Machine Translation (RBMT) is one of the earliest approaches of MT. In RBMT, human have to develop and handle rules using different grammatical conventions, lexicon [1]. One of the advantages of RBMT is that it is very simple. There are different approaches of RBMT, namely: Transfer-based RBMT, Interlingua-based RBMT, and Dictionary-based RBMT. One of the limitation of RBMT is that human need to create rules for every analysis and generation stage that proves to be quite tedious and cumbersome.

Thus, the failure of rule-based approaches led to the dominant application of corpus-based Machine Translation. It may be due to the increasing availability of machine readable text. There are different approaches of corpus-based Machine Translation namely:

1. Example-based Machine Translation
2. Statistical Machine Translation
3. Phrase based Statistical Machine Translation
4. Tree-based Statistical Machine Translation
5. Neural based Machine Translation

Example-based Machine Translation (EBMT) is one of the SMT approaches that is based on the analogy. The bilingual corpus act as its main source of knowledge base [1]. Given a new test sentence, it is translated using examples or analogy from the knowledge base. The translated sentences are then stored back into the knowledge base. This saves the effort of translating on every new test sentences. One of the limitation to this approaches is that if there is any unmatched test sentences, then it need to be regenerated from the scratch. It cannot use the concept of close phrases or neighboring word to predict the translation of unmatched words [1,10].

SMT is a data-driven or corpus-based approach. The idea of SMT comes from the information theory. It translates any document according to the probability distribution. The probability distribution applies the Bayes' Theorem.

$$\mathrm{P(m|e)=P(e|m) * P(m)} \tag{1}$$

P(e|m) in Equation (1) gives the probability that the source string is the translation of the target language. This component is called the translation model in SMT. P(m) gives the probability of getting the target string. This particular component is called the language model in SMT.

In phrase-based model of SMT, phrases are considered to be the atomic units of translation. It translates a sequences of words (or phrases) instead of word by word. The use of the phrases as a unit of translation allows the model to learn from local reordering i.e., local word orders and local agreements, idiomatic collocations, deletion and insertion by memorization that are sensitive to the contexts. The PBSMT segments the source sentences into words or phrases with uniform distribution. It then, translates each of the English phrases into Manipuri phrases according to phrase translation model estimated from the training data. Phrase based model uses the joint probability distribution P(m,e). P(m) and p(e|m) are mapped into feature of log-linear model. Finally, PBSMT needs to reorder the translated output in order to improve the fluency. Phrases longer than 3 words do not improve the quality of translations and do not follow the reordering concept has been proved in the experimental research findings of Koehn, Och, and Marcu in 2003.

The remainder section of the paper work is organized as follows: Sect. 2 describes the existing and past works on SMT and PBSMT. Section 3 explains the system architecture and methodology of the improved PBSMT system with a detailed explanation on the working of PBSMT decoder. Section 4 explains the evaluation and experimentation performs on PBSMT system. Section 5 describes the analysis result and the relative comparison between the baseline PBSMT system and the improved PBSMT system. Section 6 concludes the paper work and points the direction towards the future research works.

2 Related Works

The rapid advancement and steady progress in corpus-based machine translation reported the increasing growth of research works in the field of SMT. However, large corpora do not exist for many language pairs especially for low-level Asian languages. Thereby, we proceed our research work by making the best use of the existing parallel corpora for English to Manipuri language pair translation.

Large parallel text corpora could be mined from the web for the low density language pair has been shown in [8].

The development of phrase based SMT system trained on Europarl data has been shown in [4]. The developed system integrates the news data corpora with the translation model and language model. It results in the significant improvement of different feature weights.

A recent work on SMT incorporating syntactic and morphological processing has also been reported in [7]. The translation system has been developed for English to Hindi language pair. The developed paper work demonstrates that the baseline PBSMT system has been improved by incorporating syntactic and morphological processing with the addition of suffixes on the target side of the translation language pair.

A research paper on Manipuri-English bidirectional SMT system has also been reported. This paper work is based on factored model approach of SMT. The paper work has shown the improvement of SMT system using domain specific parallel corpora and the incorporation of morphological and dependency relations in [10].

A Manipuri to English MT system based on examples has also been reported in [10].

Incorporation of Morpho-syntactic information between the inflected form of the lemmas improves the translation quality has been proposed in [5].

The use of pivot language (English) to bridge the source and target languages in the translation process address the issue of scarcity of data resources has been proved in [11].

The unavailability of parallel corpus for many low density language is one of a big hindrances to the progress of research works. In [9], it has shown that the integration of the inflectional and derivational morphemes for Manipuri language using factor-model based SMT, solves the issue of limited availability of parallel corpora for English to Manipuri language pair translation.

Thus, the main focus of this research paper work is to build and improve the baseline PBSMT system. This requires an enormous amount of parallel corpus. But, we are limited to work with only the available text corpora. Our paper work tries to address all these issues by incorporating monolingual corpus on the target side of Manipuri language.

3 System Architecture of Improved PBSMT

There are various open source toolkits for MT system. They are Joshua, cdec, Apertium, Docent, Phrasal and so on. We use Moses toolkit for our research work [4].

This section of the paper will discuss on how we trained the system using PBSMT model. The remainder part of this section will explain the working of Phrase-based decoder and its underlying concept on how we perform the translation from English to Manipuri. The rest part of this section will discussed on how to incorporate monolingual corpus on the target side of the language pair during language model.

3.1 Architecture of PBSMT Based on SMT-Moses

SMT based Moses system is independent of any language pair. There are two main components in SMT Moses system namely: training pipeline and decoder. The training pipeline is a collection of tools and utilities (mainly written in perl and in C++). The training pipeline can accept any raw data (parallel corpora and monolingual corpus) and can train the translation model.

An important part of the translation model is language model. It is a statistical model built using monolingual corpus on the target side of the translation language pair. In our research work, we incorporate Manipuri monolingual corpus while building the language model. Moses include KenLM language model creation program lmplz. There are some freely available toolkits for language model creation namely: IRSTLM, SRILM, RandLM, OxLM, NPLM, Bilingual N-gram LM (OSM), Dependency Language Model (RDLM) and so on. In SMT, language model toolkit performs two important task: training and querying. In our work, we train the language model with KenLM, it produce an arpa file and query with blm file. The core of the translation model is the phrase translation table, that is learned from the parallel corpora. In PBSMT, training start with the alignment of the word. For this, SMT Moses use the GIZA++. Once the words are aligned, phrases pair of any length that are consistent with the aligned words could be extracted from the phrase translation table. The final step in the creation of translation model is tuning where different statistical models are weighted against each other to produce the best possible translation. Moses use the Minimum Error Rate Training (MERT) algorithm for tuning task [4]. SMT Moses system includes decoder Pharaoh which is one of a freely available tools for research purposes. Decoding in SMT Moses is a beam search process over all the possible segmentation of the phrases; translation probabilities for each phrases; and reordering cost. The decoder component in Moses tries to find the highest scoring sentence in the target language from the phrase translation table corresponding to a given source sentence. The decoder can rank the translated candidate sentences [3,4]. The basic architecture of PBSMT system is shown in Fig. 1. The methodology and control flow of the PBSMT system using Moses toolkit are explained as below:

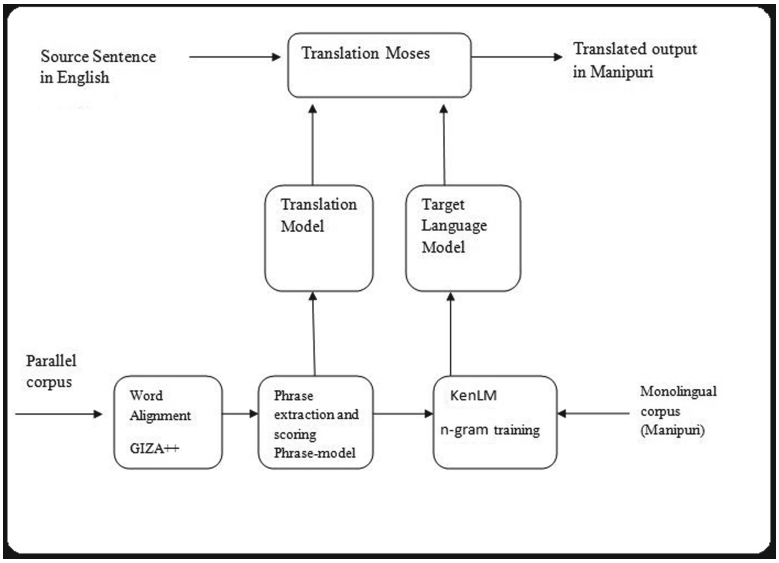

Source Sentence in English
Translation Moses
Translated output in Manipuri
Translation Model
Target Language Model
Parallel corpus
Word Alignment GIZA++
Phrase extraction and scoring Phrase-model
KenLM n-gram training
Monolingual corpus (Manipuri)

Fig. 1. System architecture of PBSMT system

1. Corpus collection and splitting the corpus for training, tuning and testing
2. Preprocessing the collected corpus
3. Language model training by incorporating monolingual corpus
4. Training the English-Manipuri translation system
5. Tuning the model and working of Phrase-based decoder
6. Running and testing PBSMT system
7. Calculating BLEU Score of PBSMT

3.2 Corpus Collection and Splitting the Corpus for Training, Tuning and Testing

This is one of the foremost tasks for any system training. We collect different parallel and monolingual text corpora from different news portal[3] and from TDIL[4] program, which is a program initiated by the Ministry of Electronics and Information Technology(MiETY), Government of India. TDIL program allows the creation and accession of multilingual resources. The collected corpus is then splitted into different sets for our system training. We collected 9565 parallel training sentences or Bitexts corpora for English to Manipuri language pair, tuning or the validation dataset consists of 1000 parallel text sentences, testing dataset consists of 100 sentences and finally, we used monolingual corpus of 1,39,411 sentences for training the language model. The information on the size and the number of sentences that we use exclusively for our system training are given in Table 1 and Table 2.

Table 1. English corpus description

Sl no	Type	Training data	Tuning data	Test data	Monolingual corpus
1	Sentences	9565	1000	100	1,39,411
2	Size	1.3 MB	141 KB	12.4 KB	26.1 MB

Table 2. Manipuri monolingual corpus description

Sl no	Type	Training data	Tuning data	Test data	Monolingual corpus
1	Sentences	9565	1000	100	1,39,411
2	Size	3.9 MB	404.6 KB	36.4 KB	26.1 MB

3.3 Preprocessing

We need to preprocess the data before training the system. For this, we tokenize, truecase and clean the data. During tokenization, a space is inserted in between the words and punctuation. MT system need to be trained on lowercase. During recasing, the initial words in each sentence is converted to their most probable lowercase character. Finally, we cleaned the data. This is done to remove all the long, misaligned and the empty sentences. Then, we limit the length of the sentences in each lines to 80 words.

[3] http://e-pao.net/.
[4] http://ildc.in/Manipuri/Mnindex.aspx.

3.4 Training the Language Model Incorporating Monolingual Corpus

Training the language model is one of foremost steps in any translation system. We have very limited parallel data. In our work, we are trying to make the best use of the available limited resources. This is achieved through the incorporation of Manipuri monolingual corpus into PBSMT system during language model training. For this, we modify the system by tuning and tweaking some of the parameters, that are used during the language model training. We used the KenLM language modeling toolkit for interpolating the monolingual corpus. It is very fast and consumes very less memory. It is compiled by default and distributed along with Moses toolkit. It use the lmplz program to estimate language models with modified Kneser-ney smoothing. The end result of language model training is an .arpa file [4]. We further query the language model by converting the .arpa file format into .blm format.

3.5 Training the English-Manipuri Translation System

We train the improved PBSMT System. Training algorithm (train-model.perl) is used by PBSMT system. The underlying control flow for training the PBSMT system are undermentioned:

3.5.1 Running GIZA++ for Word Alignment

GIZA++ is a freely available toolkit for aligning the word. The aligned words are extracted from the intersection of bidirectional runs of GIZA++ and some additional alignments point from the union of the two runs. To establish a proper alignment of the words, different heuristics approaches may be applied in [4]. Moses used the default parameter grow-diag-final parameter to run GIZA++ for aligning the word. The aligned words or phrases generated by the directional runs of GIZA++ for the translation of English to Manipuri language pair translation are shown in Table 3. The "+++" entries in the Table 3, indicates the intersection of the words or phrases during the bidirectional run of GIZA++.

Table 3. Alignment of English-Manipuri phrases using GIZA++

Sl no	মসিগী (0)	girl(1)	অসি(2)	য়াম ◌্ না(3)	beautiful(4)
this(0)	+++				
girl(1)		+++			
is(2)			+++	+++	
very(3)			+++	+++	
beautiful(4)					+++

3.5.2 Creating Lexical Translation Table

From the Phrase translation table, we can easily estimate the maximum likelihood probability p(m/e) and inverse likelihood of p(e/m) using the Noisy channel model of Bayes' theorem.

3.5.3 Extract Phrase and Score Phrase

In this, all the extracted phrases are stored into one big file. We process one phrase at a time and then compute p(e/m), for Manipuri phrase m. Similarly, p(m/e) is also estimated, for every English phrases e. Inorder to compute the scoring function, we use the lexical weighting function, word penalty, and phrase penalty.

3.5.4 Building Reordering Model

We used the distance-based reordering model. It is used to assign a cost linear to the reordered distance in the target output during language model training. For example, skipping of two word is assigned a cost higher than skipping one word. Another advanced model known as lexicalized reordering model can also be used to reorder the target output.

3.5.5 Building Generation Model

The model is built on the target side of the parallel corpus. We build the generation model for our improved PBSMT by incorporating monolingual corpus. The candidate translation C^s from the monolingual corpus of the target language can be expressed in Equation (2)

$$log((P_{LM}(C^s)) = \sum_{j=1}^{C^s} logP(C_j^s|C_1^s, C_2^s, C_3^s..........C_{j-1}^s) \tag{2}$$

3.5.6 Creating Moses Configuation File

After successfully training the model, a moses.ini configuration file is obtained. This file is used to tune the parameters further in the following subprocess.

3.6 Tuning the Model

This is one of the most time consuming tasks of the translation process. There are various tuning algorithms that are available namely: MERT, PRO, MIRA, Batch MIRA. We used the MERT algorithm (Minimum Error Rate Training) for our system training. The main idea behind tuning is to learn the weights of different discriminative model and produce the best possible translations in [4].

3.7 Mathematics of Phrase-Based Model and Working of PBSMT Decoder

Equation (3) represents the basic PBSMT equation:

$$m_{best} = \text{ arg } max_m P(m|e) = \text{arg } max_m [P(e|m)P_{LM}(m)] \tag{3}$$

Translation with the highest score is given by m_{best}. P(m|e) and $P_{LM}(m)$ are translation model and language model respectively.

$$P(\overline{e_i}|\overline{m_i}) = P(\overline{e_1}, \overline{e_2}, \overline{e_3}, \overline{e_4}, \overline{e_5}, \overline{e_6}, ...\overline{e_I}|\overline{m_1}, \overline{m_2}, \overline{m_3}, \overline{m_4}, \overline{m_5}, \overline{m_6}.....\overline{m_I}) \tag{4}$$

$$= prod_{i=1}^{I}\phi(\overline{e_i}|\overline{m_i}) * d(start_i - end_i - 1 - 1) \tag{5}$$

The LHS indicates the probability of a sequence of I phrase in the sentence m, given I phrases in sentence e. ϕ is the phrase translation probability and $d(start_i - end_i - 1 - 1)$ gives the distortion probability or the reordered distance induced by the target phrases in PBSMT system in [2].
The various control flows of Phrase-based decoder are undermentioned :

1. It extract the n-gram segments or phrases from the input source sentence.
2. It matches the extracted phrases from the phrase table.
3. It retrieved the aligned phrase along with their translation probabilities.
4. It compute the score of each hypothesis.
5. It compute the language model probabilities and distortion probabilities.
6. Lastly, future word penalty can be added so that translation do not get too long or too short. This factor indicates the difficulty in translating.

The phrase model comprises of two important components. They are Translation Phrase table from where the decoder consults how to translate the phrase and moses.ini configuration file obtained successfully after running the tunning stage which gives the updated trained weights. PBSMT decoder uses the beam search algorithm to prune the hypothesis and generate the N-best possible candidate translation [2,4]. Finally, we reorder the distance using distortion model.
Let us take an example to understand the decoding process using PBSMT.

Example 1

Input: This(0) girl(1) is(2) very(3) beautiful.(4)
Output: (PBSMT Decoder + Monolingual corpus) : মসিগী |0-0| girl |1-1| অসি য়াম ্ না |2-3| beautiful. |4-4|

From this example, it can be concluded that word or phrases in the translated output are generated from the corresponding input words or phrases.

Example 2

Input: Their(0) countries(1) wares(2) is(3) many(4) years(5) now.(6)
Output: (PBSMT Decoder + Monolingual corpus) : মখোয়গী |0-0| ্ লাজাদগী |2-2| লৈবাকশিং |1-1| অসি |3-3| চহি কয়ামরুম |4-5| now. |6-6|

This example shows that the word "countries" moved forward one step ahead in the translated output and "wares" move backward one step. The phrase "many years" is translated to the phrase " চহি কয়ামরুম" in the translated output sentence.

Finally, we perform the product of the below parameters to score the PBSMT model. They are undermentioned:

1. P_{LM}(মসিগী নুপীমচা) and P_{LM}(নুপীমচা অসি)
2. Translation Probability: P(মসিগী নুপীমচা |this girl) and P(অসি য়াম ◌্ না ফজৈ |is very beautiful)
3. Distortion Probability or the reordering distance.

Mathematically, the cost of translation in PBSMT can be expressed using Equation (6):

$$P(m|e) = \phi(e|m)^{w}eight_{phi} * LM^{w}eight_{LM} * L(e,m)^{w}eight_{d} * w(e)^{w}eight_{phi} \tag{6}$$

Accordingly, the phrase based decoder will pick the best possible translation according to the probability score and translate the corresponding source sentence to the target sentence.

3.8 Running and Testing the PBSMT System

After successfully building the PBSMT system, we can further run the PBSMT system and perform the testing operation on some new test sentences. Table 4 highlights the translated output sentences generated after successfully running and testing the phrase-based decoder.

Table 4. Running and testing PBSMT decoder

Sl no	Input test sentence	PBSMT translated output
1	You will see everything in the night	নহাক ◌্ না |0-0| উবা |1-2| পুম |3-4| ◌্ |5-5| night. |6-6|
2	I am at home in evening.	অয়ুক পুং |1-1| I |0-0| ◌্ |2-2| evening. |5-5| লৈবা |3-4|
3	What did you do last night?	What |0-0| ◌্ না লৈরম |1-1| ◌্ বা |3-3| নহাক ◌্ না |2-2| অরোইবা |4-4| night ? |5-5|
4	Have you seen me?	নহাক ◌্ না |1-1| me ? |3-3| ◌্ না |0-0| উবা |2-2|
5	Ram is a good person.	রাম |0-0| অসি য়াম ◌্ না |1-2| অফবা |3-3| person. |4-4|

3.9 Calculating BLEU Score

To calculate the BLEU Score, we used the automatic software script (multi-bleu.perl). It is observed that the improved PBSMT achieved a BLEU score of 10.15 incorporating monolingual corpus as compared to the baseline PBSMT of BLEU score 9.89 only. This shows a significant improvement. Thus, we can improve the baseline PBSMT system by incorporating monolingual corpus. This paper work addresses the issue of scarcity of parallel text corpora.

4 PBSMT System Evaluation and Experimentation

Evaluating the translated output generated from the PBSMT system is a very important concern of SMT Technology. SMT relies on two very important evaluation metrics. They are automatic metrics and human evaluation.

4.1 Automatic Evaluation

Automatic evaluation is useful in situation when we have limited amounts of parallel text corpora, less candidate test sentences and reference translations. In general, every automatic metrics should be able to correlate easily with the human evaluation. There are various automatic technique for evaluating the MT system. They are namely: BLEU, NIST, Word Error Rate, TER, GTM, METEOR and LEPOR. Among all these, we used the Bilingual Evaluation UnderStudy (BLEU) Score for our experiment.
BLEU Score is a text based metric. It is used for evaluating the quality of the translated output generated by the MT system. The main goal of BLEU Score evaluation is to compare and count the number of matches between the n-gram of the reference translation with the n-gram of the candidate translated sentence. So more the number of matches in the n-gram, closer the candidate translation is to the human translation in the citation [6]. In SMT system, calculation of BLEU score uses the sentence Brevity Penalty (BP) multiplicative factor, hyp_{len} (candidate translation sentence length) and ref_{len} (reference translation sentence length) in [6]. We assume c to be the length of the candidate translation and r be the length of the reference corpus. Equation (7) shows the computation of Brevity Penalty(BP) factor.

$$\mathrm{BP} = \begin{cases} 1 & \text{when } c > r \\ \mathrm{e}^{(1-r/c)} & \text{when } c <= r \end{cases} \tag{7}$$

Using Eq. (7) in Eq. (8), we can calculate the BLEU Score

$$BLEU = BP * exp(\textstyle\sum_{n=1}^{N} W_n log P_n) \tag{8}$$

4.2 Human Evaluation

Evaluation of the PBSMT System by humans is also equally important. Although the human evaluation may be prone to error but this cannot be neglected. This is a necessary task in order to establish a valid consistency with the obtained BLEU Score. In this evaluation, we used the adequacy-fluency strategy for the human evaluation. Table 5 show the analysis of different output generated by PBSMT and the way to assign rating to different test sentences.

Table 5. Analysis on the PBSMT translated output and the way to assign rating

Sl no	Reference Sentence	English Sentence	PBSMT	Rating
1	ঐখোয়গী অহাওবা দোসা	Our delectable dosas	ঐখোয়গী অহাওবা দোসা	VERY GOOD
2	মসিগী অশংবা টুরিজম প্রোজেক	This green tourism project	মসিগী অশংবা টুরিজম project	GOOD
3	দমপুং ফাবা অকুপ্পা মরোলশীং অমসুং এপ্লিকেসন ফোর্মেটশীংগীদমক্তা	For the complete details and application formats	তুরিষ ◌্ মপুং ◌্ অকুপ ◌্ পা অমসুং ◌্ লিকেসন থাখিবা formats	WORST
4	ভৱন অসি হৌজিককী তামিল নাদুগী মফম ২২ দা লৈ	Bhawan is present in 22 places in Tamil Nadu	ভৱন অসি হৌজিক ◌্ কী ২২ দা মফম তামিল নাদু	AVERAGE
5	য়াম্না কুইনা মতম মতমদুদা	For a long time,	মসি য়াম ◌্ না কুইনা মতম মতমদুদা,	GOOD
6	উচেক মতুনা অমসুং বারগী শউনগী লৈমায় অসিনা মসিগী হোটেল অসীগী মওং-মতৌ হেনা ফজহলি	Feather and a bar with a leather floor makes this lavish hotel a design beautiful.	উচেক মতুনা অমসুং বার ◌্ শউনগী লৈমায় অসিনা মসিগী য়াম ◌্ না হোটেল মওং-মতৌ ফজবগী.	GOOD
7	কনাগুম্বা অমনা নুমিদাংৱাইরমগী সমুদ্রা নুঙাইথোকলিঙৈদা	When one is enjoying an evening by the seaside.	মতম অসি অসিদি নুমিদাংৱাইরম অমা ◌্ সমুদ ◌্ র .	AVERAGE
8	তাকপা মনি-মুক তা লৈবা মসিগী য়ুমশারোলগী ৱায়েল অসি লর্দ কোন্নেমেরা সুইতে;	The jewel in its architectural crown is the lord Connemera suite;	তাকপা মনি-মুক ◌্ তা লৈবা মসিগী য়ুমশারোলগী ৱায়েল ◌্ অসি lord Connemera suite;	AVERAGE
9	ঐখোয়গী অহাওবা মোরোক.	Our delectable pepper.	ঐখোয়গী |0-0| অহাওবা |1-1| মোরোক. |2-2|	VERY GOOD
10	নহাক্না নুমিদাংৱাইরমদা লে◌ায়না উবা পুম্নমক ফংনি	You will see everything in the night.	নহাক ◌্ না |0-0| উবা |1-2| পুম |3-4| ◌্ |5-5| night. |6-6| |6-6|	WORST

4.2.1 Adequacy-Fluency

The adequacy factor indicates how much of the information from the original reference translation is expressed in the candidate translation. Accordingly, the evaluators assign score or rating to the test sentences. The adequacy score is also called as “fidelity” in SMT system.

Fluency indicates how natural a translated sentence sounds to the native speaker of the target Manipuri language. It involves the grammatical correction and the choice in the proper order of the word in [6].

Let us consider the following English-Manipuri translated sentence :

Example 1

Input Sentence: Tomba has a house.
Reference Sentence: য়ুম অমা লৈ
Ouput Sentence: অসি house অমা লৈ.
The PBSMT system cannot recognize the name of a person. So, Tomba is translated to "অসি", "house" is translated to "house". The translated sentence do not convey the appropriate meaning. It cannot identify the agent of the action "who bought the house?". Thus, it is not an adequate translation.
However, the word order and syntactic structure of the target grammar is maintained in the final translated output. Therefore, we can conclude that it is a fluent translation.

Example 2

Input Sentence: It is a very nice scenery.
Reference Sentence: মসি অসি য়াম্না ফজবা দ্রিশ্যা অমনী
Output Sentence: মসি ◌্ অসি য়াম ◌্ না ফজবা scenery.
The PBSMT system cannot translate the word "scenery" here. But, much of the meaning is convey by the translated sentence in this particular example. Thus, it is an adequate translation.

In addition, the syntactic structure and the word order is also preserved by the translated sentence. Therefore, the example considered above is both an adequate and a fluent translation.

Example 3

Input Sentence: This is hotel and pickles.
Reference Sentence: মসি হোটেল অমসুং আছারনী
Output Sentence: মসি |0-1| হোটেল |2-2| অমসুং |3-3| pickles. |4-4|
The PBSMT system cannot translate the word "pickles" here. But, much of the meaning is convey by the translated sentence in this particular example. Thus, it is an adequate translation.
In addition, the syntactic structure and the word order is also preserved by the translated sentence. It is also an easily readable and a fluent translated sentence. Therefore, the example considered above is also both an adequate and a fluent translation.
Example 4

Input Sentence: Konar Mess named after a community in Tamil.
Reference Sentence: কোনর মেস অসি তমীলগী খূগং অমগী মমিং লৌদুনা থোনখিবনি
Output Sentence: Konar Mess মমিং লৌদুনা থোনখিবনি ◌্ গোয়না তামিল
The PBSMT system cannot recognize the word "Konar Mess". The translated sentence is very hard to understand and correlate properly. Moreover, the translated output sentence is totally disorder. Therefore, the example considered above is not an adequate and a fluent translation.

4.3 PBSMT System Error Analysis

Analyzing the PBSMT system error is one of the main subsequent task of MT system experimentation and evaluation. We carefully examined the various types of errors generated by the PBSMT system and they are briefly undermentioned:

4.3.1 Word Order Divergence

The order of the word in English sentence is subject+verb+object. Manipuri grammar follows the word order subject+object+verb. The example discussed below show the divergence in the syntactic structure or the order of the word.

Consider the example discussed below:

Example 1

Input Sentence: Limited menu.
Translated Output Sentence: মেনু. অকক ◌্ নবা
In this, input sentence limited menu, the subject of the sentence is implicit (It). Menu refers to the object. Thus, it follows the subject+verb+object word order.

The corresponding translated output sentence in Manipuri is: "মেনু. অকক ◌্ নবা ". Therefore, it can be inferred that, the word order in Manipuri sentence is subject+object+verb.

Thus, we can infer from this example that English and Manipuri language differs significantly in their word order.
Let us consider another example below:

Example 2

Input Sentence: He is the best.
Translated output Sentence: মহাক ◌্ না |0-0| best. |3-3| অসি |1-2|
Considering the translated sentence: "he" is subject and it is translated to "মহাক ◌্ না" in Manipuri, which is a subject, "best" is object and "is" is verb and it is translated to "অসি" in Manipuri. Thus, we can conclude that word order in English language follows the subject+verb + object and the Manipuri language follows subject+object+verb word order.

4.3.2 Unable to Handle Consonant Conjunct

Manipuri language is rich in morphology and it is highly agglutinative in nature. We observe that PBSMT system cannot handle the conjunction of two consonants. It generate the consonant separately with a character ◌্ in between.

Let us consider an example given below:

Example 1

Input Sentence: Ram is a very good person.
Output Sentence: অসি য়াম্না অফবা
Here, the word "য়াম্না " (yaamna) that represent the conjunction of two consonant ম and ন cannot be represented by the PBSMT system. The two different consonant character ম and ন is separated by an UNK character ◌্ in the translated output. Instead it should be clustered together to form a single glyph ম্ন in the translated output.
Therefore, this example establishes the fact that, PBSMT system cannot handle the conjunction of two consonants.

Consider another example given below:

Example 2

Input Sentence: You have amazing of lemon.
Output Sentence: নহাক |0-0| ◌্ |1-1| ◌্ |3-3| চম ◌্ প ◌্ রা . |4-4| অঙকপা |2-2|
In this example, the word "চমপ্ " or "champra" or "lemon" represents the conjunct of two consonants প and র.. PBSMT system cannot handle the formation of glyph. It cannot generate the word "চমপ্র্". PBSMT generates the character প and র separately instead of, combining these two character together to form a single character conjunct প্র্ in the final translated output sentence. Therefore, PBSMT system cannot handle the conjunction of consonant. This need to be solved.

Let us consider another example in different scenario:

Example 3

Input Sentence: Limited menu.
Output Sentence: মেনু. অকক ◌্ নবা
In the example discussed above, the PBSMT system cannot handle the conjunction of two similar consonants ক and ক to produce a glyph "ক্ক". PBSMT cannot translate the word "অকক্কনবা (limited)". This is also one of the limitations of PBSMT system. It need to be handled.

4.3.3 Unable to Handle Case Markers and Suffixation

The PBSMT system cannot handle the suffixation of suffix or case markers on the target side of the translated output sentences. It cannot identify the agent or the doers of the action in the final translated sentence. For example, the suffixes -ni নী, -dbni দবনী, -ri রী, re রে, (-oi/- ওই) and so on, cannot be suffixed by the system to construct a perfect translation of the input sentences.
Let us consider another example.

Example 1

Input Sentence: He like double malai.
Output Sentence: মহাক ◌্ না |0-0| দবল |2-2| malai. |3-3| ◌্ অমগুম ◌্ না |1-1|
The translated output sentence do not clarify who like "double malai". Therefore, the PBSMT system cannot identify the agent of the action in the translated sentence.
Let us consider another example.

Example 2

Input Sentence: This is hotel and pickles.
Output Sentence: মসি |0-1| হোটেল |2-2| অমসুং |3-3| আছার. |4-4|
The output translated sentence would be a perfect and a complete sentence with 100 % adequacy and 100 % fluency, if the PBSMT system were able to affix the suffix-ni (-নী) to the translated sentence. Thus, we can conclude that PBSMT system cannot handle the affixation of suffix to the translated output sentences.

4.3.4 Simple and Compound Sentence

The study of Statistical Machine Translation from English to Manipuri Language pair translation infers that simple sentence are translated without error as compared to the complex sentence.

The translation of simple sentence given below illustrates the above concept.
Example 1

Input Simple Sentence: Last edited
Output Simple Sentence: অকোনবা শেমদোক
Each of the word in the source sentence is translated to their corresponding word in the target language. Therefore, PBSMT system can translate the simple sentence quite easily.
Example 2

Input compound Sentence: We enjoy hot chicken and mutton food.
Output compound Sentence: ঐখোয়না |0-0| নুংঙাইনা |1-1| অশাবা |2-2| য়েন |3-3| অমসুং য়াও

Each of the word in the source sentence is translated to their corresponding word in the target language. Only the meaning of the word "food" cannot be translated by the PBSMT system. Therefore, PBSMT system can translate the compound sentence to a great extent.
Example 3

Input complex Sentence: Eat straight from the plantain leaf with a wide range of pickles to accompany.
Output complex Sentence: চাবগী ওইবা লমকোই খোঙচত ◌্ ◌্ লাদা মনাদা অমা মখল য়াম ◌্
We can see the difference from the above considered two example. PBSMT system translate the complex or the long sentence with the the character ◌্ in between the alphabet or the character. We can see there are 5 ◌্ character in the translated output. It is quite difficult to convey the meaning of the translation in the complex sentence even. Therefore, PBSMT cannot handle the complex sentence correctly.

4.3.5 Unable to Translate Some Words

The study of Statistical Machine Translation from English to Manipuri language pair shows that some words are not able to translate by the PBSMT System. The corresponding target words need to be transliterated further from the lexicon. The translation given below illustrates the above concept.
Example 1

Input Sentence: What rubbish!
Output Sentence: করি rubbish! rubbish!
Here the words "rubbish" cannot be translated even though it is a simple sentence.

Example 2

Input Sentence: This girl is very beautiful.
Output Sentence: মসিগী girl অসি য়াম ◌্ না beautiful.
Here the words "beautiful" and "girl" cannot be translated by the PBSMT System.
Example 3

Input Simple Sentence: First aid kit and manual
Output Simple Sentence: অহানবা হীদাকশিং অমসুং manual
Here the word "manual" cannot be translated by the PBSMT System.

5 Result Analysis and System Comparison

1. Analysis on the Basis of Inter-Rater Reliability or Agreement:

In this experimentation, we evaluate and compare the performance of two different system. The first system S1 is the baseline PBSMT system and system S2 is the improved PBSMT system build incorporating monolingual corpus. We calculate the Percentage(%) Agreement/Inter-Rater Reliability from the rating assigned by different evaluators. We represent the calculated Inter-Rater Reliability along the Y-axis of the graph and the agreement between the evaluator along the X-axis. This is depicted clearly in the figure. From Fig. 2, it is observed that the Percentage(%) Agreement/Inter-Rater Reliability between different evaluators converges more in System 2 which is indicated by the UPPER line for the improved PBSMT system. It has been observed that there is much more convergence in the opinion of different evaluators while evaluating the improved PBSMT system as compared to the baseline PBSMT.

2. Analysis on the Basis of BLEU Score:
In the next analysis, we used the automatic metrics BLEU Score to evaluate the performance of two different system. Moses system uses the script multi-bleu.perl to generate the BLEU Score of the developed two different systems. It can be observed from Fig. 3, that the improved PBSMT system achieved a BLEU Score of 10.15 in comparison to the baseline PBSMT system of BLEU Score 9.89 using the same training, tuning and testing datasets. The difference in the improved BLEU Score by a factor of 26% shows that a marked significant achievement by the System S2 as compared to System S1. From these research findings, we can conclude that more the BLEU score the more is the convergence to the Inter-Rater Agreement among the evaluators.

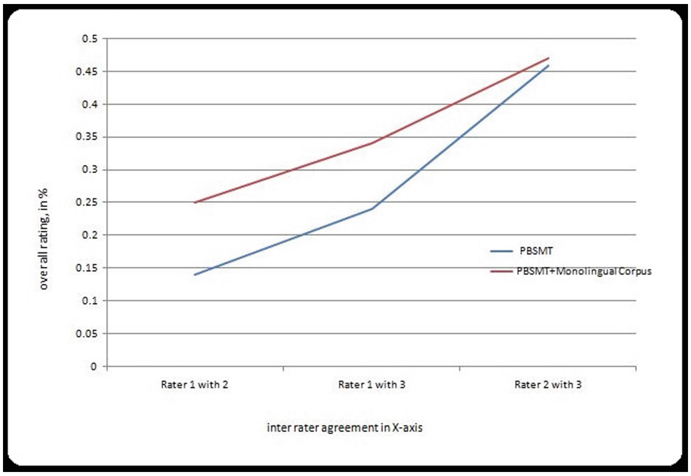


Fig. 2. System performance comparison based on evaluator Inter-Rater reliability

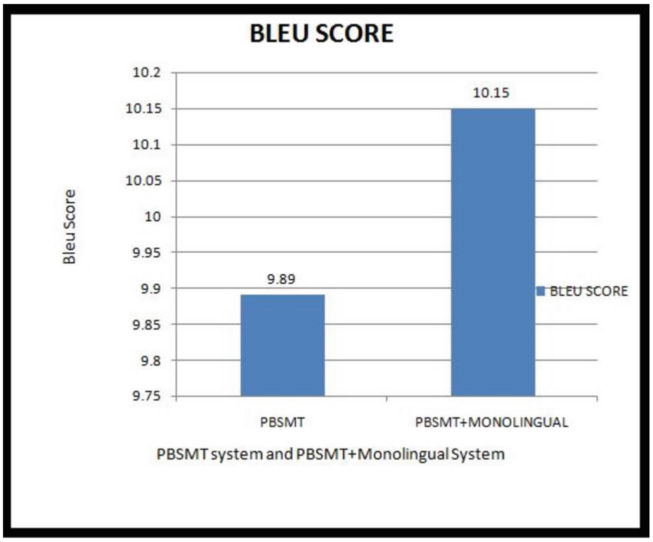


Fig. 3. System performance comparison based on BLEU score

Table 6. Mean of the overall rating for adequacy and fluency

Sl no	System	Evaluator	Adequacy	Fluency
1	PBSMT Baseline	Evaluator1	1.73	1.56
2	PBSMT Baseline	Evaluator2	2.01	1.99
3	PBSMT Baseline	Evaluator3	2.48	2.32
4	PBSMT+Monolingual	Evaluator1	1.86	1.89
5	PBSMT+Monolingual	Evaluator2	2.13	2.07
6	PBSMT+Monolingual	Evaluator3	2.52	2.46

3. Analysis on PBSMT Translated Output:

   In the last part of analysis section, we have conducted some comparison with respect to fluency on different output translated by the baseline PBSMT system and the improved (PBSMT+Monolingual corpus) system. The different parameters has been evaluated from various feedback posted by different evaluators during the evaluation session.
   Let us consider an example below to illustrate the above concept:
   Input Sentence : What rubbish!
   Output sentence (System S2 PBSMT+monolingual corpus) : করি (What) rubbish!
   Output Sentence (System S1 Baseline PBSMT) : মসিগী (This) rubbish!
   It can be seen clearly from the example that system S2 improves the fluency and readability of the translated output sentence as compared to the output

sentence translated by the baseline PBSMT system S1. The native speakers find the translated word “করি” to be more appropriate and suitable in this particular context as compared to the translated word “মসিগী”, which is translated by the system S1.
Thus, the system S2 (PBSMT+monolingual corpus) outperforms the baseline PBSMT System. From this observation, it can be concluded that system S2 improves the fluency and readability of the translated output. Table 6 shows the mean of the overall rating of adequacy and fluency rated by the evaluators.

6 Conclusion and Future Works

The study on this research paper work clearly indicates that the interpolation of monolingual corpus on the target side of the English to Manipuri language pair translation significantly improves the BLEU Score and also the fluency of the translated output sentences. The research work conducted on SMT using Phrase-based approach could be more improved with more training data sets. However, we perform PBSMT system training by making the best use of the limited available resources. This paper work addresses the issue of the unavailability of the parallel text corpora. This is accomplished through the incorporation of Manipuri monolingual corpus. We interpolated monolingual corpus during the language model training. It achieves an improved BLEU Score of 10.15 as compared to the baseline PBSMT system of 9.89 only. Besides these, from the feedback and suggestion posted by different evaluators, the fluency and quality of the translated output improves drastically. This is one of the major contribution to our research findings on PBSMT system. Another point worthwhile to be mentioned, output of the translated output and even the BLEU Score could be enhanced further. This would have been achieved through the incorporation of more morphological information and case markers on the target side of the translation language pair in the citation [9]. We could still focus on the post editing task to handle the system error and after the correction of these translated output sentences, we could feed the correct sentences during retraining the system. Thereby, PBSMT system could be trained to learn more correctly and able to predict accurately on the new test sentences.

Acknowledgments. I would like to express my deepest appreciation to the Technology Development for Indian Languages (TDIL) Programme, initiated by the Ministry of Electronics and Information Technology, Govt. of India for sharing the valuable parallel corpus on English to Manipuri Language pair and the monolingual corpus in Manipuri Language for this research paper. Furthermore, I would like to extend my heart full gratitude to the Department of Computer Science and Engineering, National Institute of Technology, Mizoram for providing me the required financial assistance and the laboratory facilities for conducting out the full experimental research works on this research paper.

References

1. Antony, P.: Machine translation approaches and survey for Indian languages. Int. J. Comput. Linguist. Chin. Lang. Process. **18**(1), 47–78 (2013)
2. Dave, S., Parikh, J., Bhattacharyya, P.: Interlingua-based English-Hindi machine translation and language divergence. Mach. Transl. **16**(4), 251–304 (2001)
3. Hoang, H., Koehn, P.: Design of the moses decoder for statistical machine translation. In: Software Engineering, Testing, and Quality Assurance for Natural Language Processing, pp. 58–65. Association for Computational Linguistics (2008)
4. Koehn, P.: Machine Translation System User Manual and Code Guide (2011)
5. Nießen, S., Ney, H.: Statistical machine translation with scarce resources using morpho-syntactic information. Comput. Linguist. **30**(2), 181–204 (2004)
6. 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, pp. 311–318. Association for Computational Linguistics (2002)
7. Ramanathan, A., Hegde, J., Shah, R.M., Bhattacharyya, P., Sasikumar, M.: Simple syntactic and morphological processing can help English-Hindi statistical machine translation. In: IJCNLP, pp. 513–520 (2008)
8. Resnik, P., Smith, N.A.: The web as a parallel corpus. Comput. Linguist. **29**(3), 349–380 (2003)
9. Singh, T.D.: Addressing some issues of data sparsity towards improving English-Manipuri SMT using morphological information. Monolingual Machine Translation p. 46
10. Singh, T.D., Bandyopadhyay, S.: Manipuri-English example based machine translation system. Int. J. Comput. Linguist. Appl. (IJCLA), ISSN pp. 0976–0962 (2010)
11. Utiyama, M., Isahara, H.: A comparison of pivot methods for phrase-based statistical machine translation. In: HLT-NAACL, pp. 484–491 (2007)

On the Influence of Machine Translation on Language Origin Obfuscation

Benjamin Murauer(✉), Michael Tschuggnall, and Günther Specht

Department of Computer Science, University of Innsbruck, Innsbruck, Austria
b.murauer@posteo.de, michael.tschuggnall@ches.pro, guenther.specht@uibk.ac.at

Abstract. In the last decade, machine translation has become a popular means to deal with multilingual digital content. By providing higher quality translations, obfuscating the source language of a text becomes more attractive. In this paper, we analyze the ability to detect the source language from the translated output of two widely used commercial machine translation systems by utilizing machine-learning algorithms with basic textual features like n-grams. Evaluations show that the source language can be reconstructed with high accuracy for documents that contain a sufficient amount of translated text. In addition, we analyze how the document size influences the performance of the prediction, as well as how limiting the set of possible source languages improves the classification accuracy.

Keywords: Text classification · Machine translation detection · Original language identification

1 Introduction

The amount of textual data being shared through various channels over the Internet is growing constantly. Considering the fact that this data is transferred across cultural as well as political borders, machine-translation is an often used means in case the source or target language is understood insufficiently. On the other hand, the growing quality of publicly available services from, e.g., Google or Microsoft, can also be used to obfuscate the source language, and thus the geographical origin. While there exist many approaches in the field of stylometry to identify the author of documents [17] (which is also frequently used in digital forensics [13]), to determine the native language of a person based on samples written in a non-native language (Native Language Identification, e.g., [8]), or to detect the translation tool of a machine-translated text [1] it has not been investigated whether similar techniques can be utilized to predict the source language of machine-translated text with satisfying accuracy. Algorithms which are able to reconstruct the original language could be utilized for, e.g., detecting plagiarism: once a part of a document is confirmed to be machine-translated and the original language is known, further investigations can focus on that languages. Regarding forensics, a use case could include determining the native language of a threat that has been automatically translated, e.g., to restrict the geographical origin and consequently the amount of possible authors.

We analyze the prediction performance of commonly used stylometric features such as n-grams – which are known to work well for various text classfication problems [8,12] – regarding the detection of the source language from machine-translated text. Adhering to the fact that most of online text is shared in English [14], we restrict the target language to be English as well for this study.

Analogous to the field of authorship attribution where it has been shown that an increasing amount of possible authors decreases the prediction performance, we additionally analyze the influence of the number of possible source languages on the quality of the prediction. Finally, we determine the amount of text that is required to achieve a specific accuracy for predicting the source language.

Concretely, we answer two main research questions (RQ) in this paper:

RQ1: Given a document, can it be detected whether it was machine-translated, and if yes, how accurate can the source language be determined?
RQ2: How do different feature sets, varying amounts of possible source languages and document lengths influence the prediction accuracy?

We find that using traditional features, the original source language of a translated text can be predicted with a very high accuracy. Additionally, we show that this prediction is possible with even very small amounts of text for many possible languages.

2 Related Work

Previous work in authorship attribution [5,16] has shown that the use of automatic translation techniques actually adds information to a text document since the translation itself adds details specific to the methods used. We assume this fact is still valid although the engine used by the translation services has changed from statistical methods to neural networks[1,2].

A field of research closely related to the problem presented in this paper is Native Language Identification (NLI), which aims to find the native language of authors given only documents in a foreign language. Features used for this classification task range from traditional n-grams [2,11] over string kernels [7] to detailed stylistic analyses of the words and grammar that was used by the authors [3,4,18,19].

In an approach more closely related to the problem addressed by this paper, Koppel et al. determine the source language of human-translated documents by analyzing various types of errors introduced by non-native authors [10].

The effect of automatic machine translation on text has been analyzed with focus on different aspects. Detecting the tools which have been used to translate a text has been shown to be already possible for very small amounts of text [1]. Caliskan and Greenstadt show that repeated translation of text does not obfuscate an author's individual writing style [5]. Instead, the translation itself adds even more information to the text, making

[1] https://www.blog.google/products/translate/found-translation-more-accurate-fluent-sentences-google-translate/.
[2] https://blogs.msdn.microsoft.com/translation/2016/11/15/microsoft-translator-launching-neural-network-based-translations-for-all-its-speech-languages/.

it possible to determine which translation engine was used. In their research, a text that has been translated two and three times is shown to still contain enough information to determine the authorship with an accuracy of over 90%. Similarly, it has been shown that the effectiveness of using machine-translation to obfuscate the original text and thus reduce the accuracy of authorship verification is limited [15].

3 Methodology

At a glance, we utilize textual features commonly used in stylometry to quantify the characteristics of translated English text originating from a specific source language. These features are then used to train state-of-the-art machine-learning classifiers, which are finally used to predict the source language.

3.1 Dataset

We used two different sources to construct a representative dataset containing German (de), English (en), Spanish (es), French (fr), Italian (it), Japanese (ja), Korean (ko), Dutch (nl), Turkish (tr) and Chinese (zh) text: firstly, the Leipzig Corpus Collection (further called *lcc*) [6] contains random sentences from news articles, web pages and Wikipedia articles. Secondly, random Wikipedia articles were selected for each of the languages mentioned above, which will collectively be referred to as *wiki*.

For each language from both *lcc* and *wiki*, 5,000 sentences were chosen randomly and translated to English using the services of Google[3] and Microsoft[4]. Subsequently, 9 of the 10 languages are translated, whereas the original English sentences are not translated. The dataset generation process is displayed on Fig. 1, where in the first step the sentences from both datasets are combined, yielding 10,000 sentences for each language.

Thereby, the translation itself was performed sentence-wise. Both translation engines feature an automatic detection of the source language, however, since this information was known it was provided to them. This implicitly enables our prediction model to detect whether a document has been machine-translated at all. For all further investigations, the texts from the two services were kept separate, i.e., the computed models were tested on documents coming from the same translation service that was also used for training. While the performance of cross-translation-service evaluations might be of interest at a first glance, this was omitted since it has already been shown that detecting the tool which was used to translate a text can be detected easily [1], i.e., in a realistic scenario one would probably use such an approach as a preliminary step to choose the correct model.

Document Sampling. To answer parts of RQ2, i.e., the influence of text length on the source language prediction quality, documents of various text sizes are required. To construct expressive documents, the translated sentences were combined randomly to

[3] https://translate.google.com.
[4] https://www.microsoft.com/en-us/translator/.

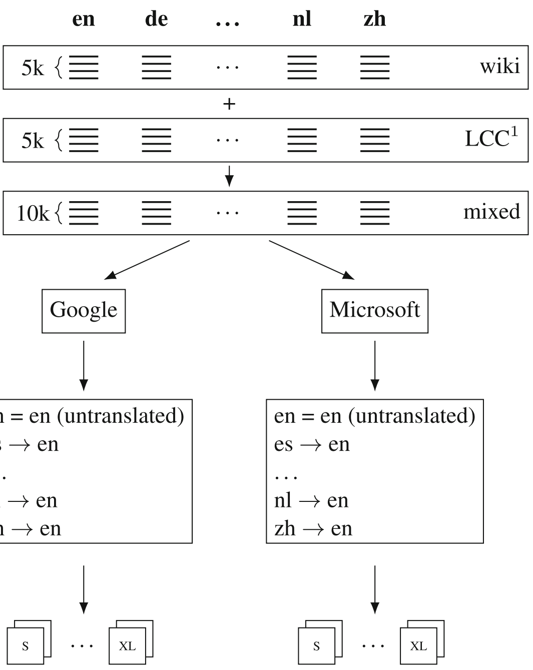


Fig. 1. Dataset generation workflow

create documents of various sizes, i.e., data sets consisting of short (S), medium (M), long (L) and very long (XL) documents, respectively[5]. The outcome of this step of the workflow is depicted in the bottom part of Fig. 1, and details about the resulting datasets are listed in Table 1.

The choice for the document sizes is thereby loosely based on typical sizes of online texts according to social media statistics[6] All documents created are unique with respect to their sentences contained, i.e., one sentence can only be part of one document per size class. This limitation causes the data set with the longest documents (XL) to contain less documents than the other datasets, for which a maximum of 1,000 documents were created.

3.2 Features

To distinguish the origin of the translated documents, several text features are used, which are listed in Table 2. These features were calculated by the Python library scikit-learn[7] as well as the language processing library spaCy[8] for POS tagging. Regarding

[5] The sampling on a random basis was required since the *lcc* dataset only contains random sentences, rather than cohesive documents.

[6] e.g., https://blog.bufferapp.com/optimal-length-social-media.

[7] http://scikit-learn.org/.

[8] https://spacy.io/.

Table 1. Data sets with varying document lengths

Length	No. of words	No. of docs
short (S)	5–50	1,000
medium (M)	51–200	1,000
long (L)	201–1,000	1,000
xlong (XL)	$\geq$ 1,001	312 (on avg)

the features used, the punctuation occurrence counts the amount of selected punctuation marks[9] in each document. Features 2 and 3 count the occurrences of distinct words, separated by whitespace, whereby feature 3 disregards all English stop words[10]. Features 4–23 count the occurrences of n-grams of various lengths of characters and POS-tags, respectively. The n-gram-features denote the combination of all n-grams ranging from length 1 to 4. The *tf-idf* term denotes that the respective measure is normalized using term frequency-inverse document frequency metric provided by scikit-learn[11].

Combinations of features were tested, but provided no additional accuracy over using single features. This is displayed in Sect. 4.

Table 2. List of utilized features

Id	Feature name
1	Punctuation occurrence
2	Word occurrence
3	Word occurrence, w/o stop words
4–8	Character 1, 2, 3, 4, n-grams
9–13	Character 1, 2, 3, 4, n-grams w. tf-idf
14–18	POS-tag 1, 2, 3, 4, n-grams
19–23	POS-tag 1, 2, 3, 4, n-grams w. tf-idf

3.3 Classifier

In order to determine the classifier for this task, a preliminary evaluation has been conducted on the S-dataset, using all features described in Table 2 individually. Thereby, several popular classifiers were tested: Naive Bayes, K Nearest Neighbors (KNN), Support Vector Machines (SVM) with Linear and RBF Kernels, Random Forest and a Multilayer Perceptron (MLP). This preliminary step was required due to the infeasibility of testing all classifiers with all experimental runs described in Sect. 4. To estimate the

[9] `: ; . , ? ! " # $ % & ( ) [ ] { } * + - \ / ^ _ ` | ~`.

[10] We rely on the list of stop words provided by scikit-learn.

[11] http://scikit-learn.org/stable/modules/feature_extraction.html#tfidf-term-weighting.

classifier performances, the default parameters given by the *scikit-learn* library have been used. The results of this preliminary run are displayed in Table 3. We observe that the linear SVM outperformed all other classifiers, if only by a small margin. The neural network (MLP) has a similar performance, at the cost of a much longer runtime. Being in line with other research in this area [9, 10] and adhering to the results of this test, the linear SVM was used for further experiments.

Table 3. Accuracy of different classifiers on the short data set. The feature ids correspond to the features listed in Table 2. The best performing feature (with id = 13) corresponds to character n-grams with tf-idf normalization.

Classifier	Microsoft		Google	
	Score	Feature id	Score	Feature id
Naive Bayes	0.39	12	0.31	12
KNN	0.30	11	0.24	12
SVM (linear)	**0.44**	13	**0.34**	13
SVM (RBF)	0.19	14	0.18	14
Random Forest	0.28	3	0.21	3
MLP	**0.44**	13	**0.34**	13

3.4 Experiment Setup

As argued previously, the texts translated by the two translation services have not been mixed for all following calculations, i.e., we never test documents translated by Google on models that have only seen documents translated by Microsoft and vice versa.

To answer both RQs, two separate experiments were conducted: In the first setup, the influence of the number of possible source languages on the resulting accuracy is analyzed. At first, a classification process has been performed on the *short* documents, including all source languages. Subsequently, each combination of 2, 3 and 4 possible source languages has been selected, whereby each combination also contains original, untranslated English documents. The restriction that the combinations must contain the English documents ensures that each combination also contains a non-translated set of documents. This way, any difference in classifying translated vs. original text can be determined as well.

The second experiment subsequently analyzes the influence of the document length on the classification performance. Here, all possible languages are taken into account, resulting in a 10-class classification problem.

4 Results

To quantify the individual expressiveness of each feature, we first conducted isolated evaluations for each feature using the S-dataset. The corresponding results can be found in Table 4. For readability reasons, only selected features are shown in the figure, with the best performing feature printed in bold face. Then, the best performing features were selected, and combinations thereof were tested. Results of these experiments are displayed in Table 5, which displays the best performing feature set (character n-grams with tf-idf normalization, i.e., utilizing combined character 1-, 2-, 3- and 4-grams) including other well-performing combinations. Comparing these results to the previous single-feature evaluation of Table 4, it can be seen that combining features (sets) does not improve the classification performance.

Table 4. Accuracy of all single features

Feature id	1	2	3	4	5	6	7	8	9	10	11	12
Microsoft	0.21	0.36	0.33	0.25	0.35	0.39	0.39	0.41	0.29	0.36	0.40	0.41
Google	0.15	0.28	0.25	0.21	0.28	0.29	0.30	0.33	0.23	0.29	0.30	0.31
Feature id	13	14	15	16	17	18	19	20	21	22	23	
Microsoft	**0.44**	0.20	0.27	0.25	0.22	0.27	0.20	0.27	0.25	0.21	0.24	
Google	**0.34**	0.18	0.24	0.24	0.22	0.26	0.18	0.24	0.23	0.22	0.24	

Table 5. Accuracy of best feature and selected combinations

Features used	accuracy	
	Microsoft	Google
char-n (tfidf)	**0.44**	**0.34**
char-n (tfidf) + POS-n (tfidf)	0.25	0.24
char-n (tfidf) + char-n	**0.44**	**0.34**
char-n (tfidf) + char-4 (tfidf)	0.42	0.34
char-n (tfidf) + char-4	0.41	0.30

Table 6. Part of the confusion matrix for short documents translated by Google

	de	en	es	nl
de	57	24	23	39
en	7	178	4	11
es	28	7	86	17
nl	26	15	26	72

The overall performance on classifying the original language of a machine translated text is high. Figure 2 shows the classification performance with a varying amount of possible source languages. It can be seen that for all text lengths tested, the prediction is well above the baseline, which represents random guessing out of the possible languages. In case of only two possible source languages, text translated by Microsoft can be classified with an accuracy of over 0.85 when having only short documents as input. The combination of character 1- to 4-grams with tf-idf normalization proves to be the best suiting feature set for both the Microsoft and Google translations.

As expected and displayed in Fig. 3, the quality of the predictions also increases substantially with the amount of text available. For the XL-dataset, the accuracy of the classifier reaches values of over 0.99. This result suggests that if the amount of text translated by an engine is large enough, i.e., more than 1,000 words, its unique properties can be used to predict the source language nearly perfectly.

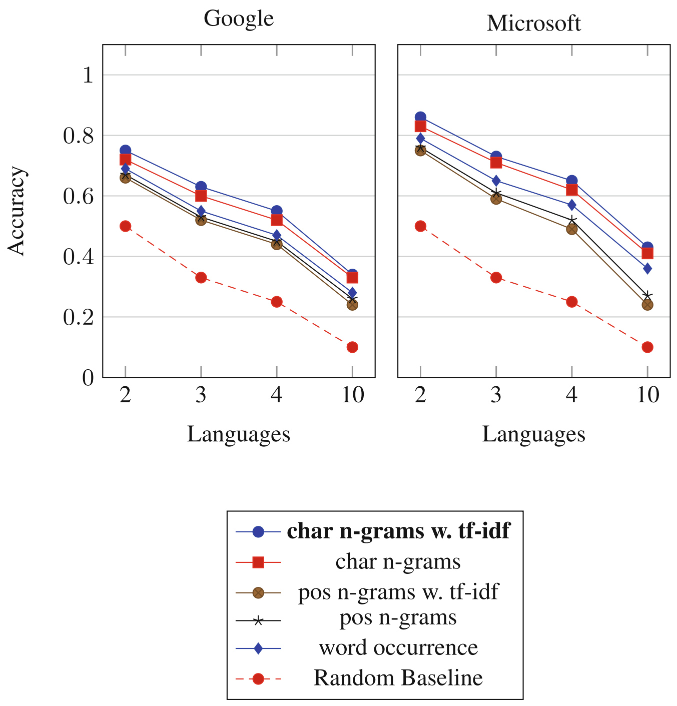


Fig. 2. Varying number of possible languages using short document lengths

Both experiments show that character n-grams with tf-idf normalization achieve the best performance, regardless of the translation engine used. Another fact that can be extracted from the results is that the classification performs worse on text translated by Google rather than by Microsoft. It suggests that Google is currently able to transfer more complex linguistic structures onto its translation than Microsoft does. Note that in general all results are based on the technology currently used by the two translation engines, which may change in the future. While it can be assumed that the prediction

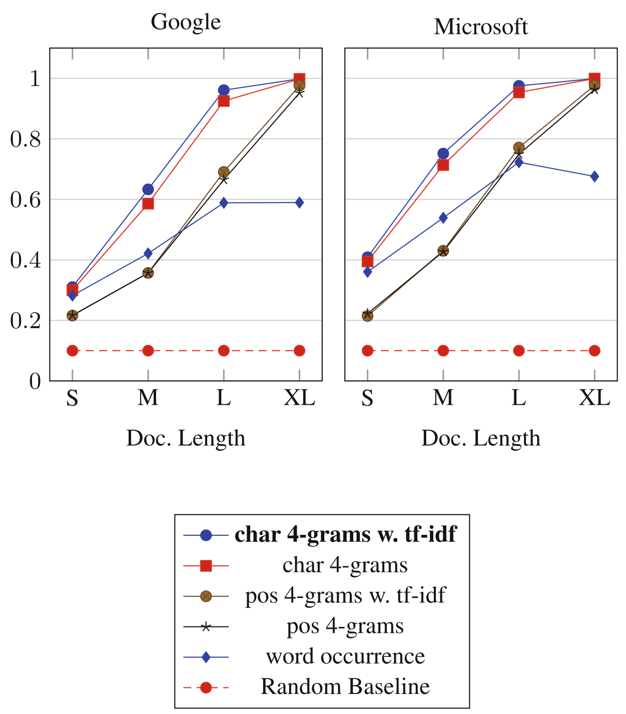


Fig. 3. Varying document lengths using 10 languages

accuracy decreases with improvements of the tools, future work should investigate this problem in detail.

Table 6 shows an excerpt of the confusion matrix of short documents using the Google translation service. Due to the origin of the languages it seems intuitive that some pairs (e.g., es–en) are easier distinguishable than others (e.g., nl–de), which is also reenforced by our experiments. However and unfortunately, the SVM used for the classification provides no information about the relevance of the single character n-grams features, so we can't argue which parts of the translated text indicates the original language at this point. Note that while other classifiers indeed can provide such information (e.g., decision trees or naive Bayes), their prediction accuracy is significantly worse, making statements about the indicators unreliable.

Answering RQ1, the experiments clearly show that it can be determined whether an English text is machine-translated, and that the source language can be predicted with high accuracy. Moreover, the prediction quality increases significantly by restricting the amount of possible source languages as well as by providing longer texts (RQ2).

5 Conclusion and Future Work

In this paper, we have shown that, given an English text, it can be determined if it was translated by using stylometric features in combination with machine-learning. Moreover, if translation was applied, the source language can be predicted at a high accuracy for regular length texts, and at a near-perfect accuracy when providing longer text samples. If the number of possible source languages is low, the prediction is reliable even for short length documents. Possible extensions of this study include considering more different language classes (e.g., Arabic and African), analyzing other target languages for the translation or finding optimal parameters for different classifiers (e.g., Neural Networks). However, while we showed that source language detection is possible at high accuracy in general, it remains unclear what the discriminating features (i.e., n-grams) are for the respective languages. Using different classifiers which contain feature importances could be used to gain an intuitive understanding of the differences of specific language pairs. Other approaches can include automatic measures like recursive feature elimination, which was omitted due to a limited timeframe. Different kinds of features, such as grammar based features, may help to gain further understanding of the translation systems analyzed, which can subsequenly be leveraged to increase the performance of the model ever further.

References

1. Aharoni, R., Koppel, M., Goldberg, Y.: Automatic detection of machine translated text and translation quality estimation. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, vol. 2, Baltimore, USA, June 2014, pp. 289–295. Association for Computational Linguistics (2014)
2. Bykh, S., Meurers, D.: Native language identification using recurring N-grams - investigating abstraction and domain dependence. In: Proceedings of the 24th International Conference on Computational Linguistics (COLING 2012),Mumbai, India, pp. 425–440, December 2012
3. Bykh, S., Meurers, D.: Exploring syntactic features for native language identification: a variationist perspective on feature encoding and ensemble optimization. In: Proceedings of the 25th International Conference on Computational Linguistics (COLING 2014), Dublin, Ireland, August 2014, pp. 1962–1973 (2017)
4. Bykh, S., Meurers, D.: Advancing linguistic features and insights by label-informed feature grouping: an exploration in the context of native language identification. In: Proceedings of the 26th International Conference on Computational Linguistics (COLING 2016), pp. 739–749, December 2016
5. Caliskan, A., Greenstadt, R.: Translate once, translate twice, translate thrice and attribute: identifying authors and machine translation tools in translated text. In: Proceedings of the 6th International Conference on Semantic Computing (ICSC 2012), pp. 121–125. IEEE, September 2012
6. Goldhahn, D., Eckart, T., Quasthoff, U.: Building large monolingual dictionaries at the Leipzig corpora collection: from 100 to 200 languages. In: Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC 2012) (2012)
7. Ionescu, R.T., Popescu, M.: Can string kernels pass the test of time in Native Language Identification? In: Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications, pp. 224–234. Association for Computational Linguistics, September 2017

8. Ionescu, R.T., Popescu, M., Cahill, A.: Can characters reveal your native language? A language-independent approach to native language identification. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014), pp. 1363–1373. Association for Computational Linguistics, October 2014
9. Joachims, T.: Text categorization with support vector machines: learning with many relevant features. In: Nédellec, C., Rouveirol, C. (eds.) ECML 1998. LNCS, vol. 1398, pp. 137–142. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0026683
10. Koppel, M., Schler, J., Zigdon, K.: Automatically determining an anonymous author's native language. In: Kantor, P., et al. (eds.) ISI 2005. LNCS, vol. 3495, pp. 209–217. Springer, Heidelberg (2005). https://doi.org/10.1007/11427995_17
11. Malmasi, S., Dras, M.: Language transfer hypotheses with linear SVM weights. In: Proceedings of the 2014 Conference on Empirical Methods in Natuarl Language Processing (EMNLP 2014), pp. 1385–1390. Association for Computational Linguistics, October 2014
12. Malmasi,S., et al.: A report on the 2017 native language identification shared task. In: Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications, pp. 62–75. Association for Computational Linguistics, September 2017
13. Nirkhi, S., Dharaskar, R.V.: Comparative study of authorship identification techniques for cyber forensics analysis. Int. J. Adv. Comput. Sci. Appl. **4**(5) (2013)
14. Pimienta, D., Prado, D., Blanco, A.: Twelve years of measuring linguistic diversity in the internet: balance and perspectives (2009)
15. Potthast, M., Hagen, M., Stein, B.: Author obfuscation: attacking the state of the art in authorship verification. In: Working Notes Papers of the CLEF 2016 Evaluation Labs, CEUR Workshop Proceedings. CLEF and CEUR-WS.org, September 2016
16. Rao, J.R., Rohatgi, P.: Can pseudonymity really guarantee privacy? In: Proceedings of the 9th Conference on USENIX Security Symposium, volume 9 of SSYM 2000, pp. 7–7. USENIX Association, August 2000
17. Stamatatos, E.: A survey of modern authorship attribution methods. J. Am. Soc. Inform. Sci. Technol. **60**(3), 538–556 (2009)
18. Swanson, B., Charniak, E.: Extracting the native language signal for second language acquisition. In: Proceedings of NAACL-HLT, pp. 85–94. Association for Computational Linguistics, June 2013
19. Swanson, B., Charniak, E.: Data driven language transfer hypotheses. In: Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, volume 2: short papers, Gothenburg, Sweden, April 2014, pp. 169–173. Association for Computational Linguistics (2014)

Error Classification and Evaluation of Machine Translation Evaluation Metrics for Hindi as a Target Language

Samiksha Tripathi(✉) and Vineet Kansal

AKTU, Lucknow 226031, UP, India
Samiksha.tripathi@gmail.com

Abstract. Evaluation of Machine Translation (MT) is an onerous but a critical task. Automatic evaluation metrics evaluates the adequacy and fluency of a translated sentence. Automatic evaluation of machine translation is able to compare between two different translation systems but it doesn't provide any insights into the kind of errors a translation system is making. Our error classification, inspired by Vilar et al., has extended categories more linguistically for Hindi language. In this paper, we will explore various evaluation metrics for machine translation and perform extensive linguistic and statistical analysis of the translation output to identify primary issues in existing framework of automated metrics for English-to-Hindi MT systems. This leads us to better insights for improvement of these metrics for English-to-Hindi automatic machine translation.

Keywords: Evaluation metric · English-to-Hindi MT · Error classification in MT

1 Introduction

In Machine Translation, meaning of a text in source language is fully transformed to the equivalent meaning in target language. Translation can never be word for word substitution due to language specific characteristics, so it is a very challenging exercise to evaluate the Machine Translation. Most of the evaluation metrics focus on lexicon matching between the tokens of candidate translation (produced by the MT system) and reference translation [1].

There are many evaluation metrics being developed. These metrics work accurately for many languages but they fail in rating translation correctly for morphologically rich target languages such as Hindi. It is important to capture various types of divergence between English and Hindi and then try to evaluate against a specific evaluation metric system. We first did a linguistic error analysis to get qualitative insights into the variety of issues that crop up for English-to-Hindi machine translation. Subsequently a detailed statistical evaluation is performed to quantify the problems encountered with existing framework of MT evaluation metrics. Adequacy (Comprehensiveness) and Fluency (Naturalness) are most desirable features for correct translation [1]. A translation can be defined as adequate if it preserves the meaning of the source language and does not add additional information. Grammatical and natural text in the target language is known as fluency. Adequacy and fluency are the major aspects of the machine translation performance.

Evaluation of automated translation system can be done using Post Editing (PE) which makes correction in target language text that has been translated from source language using automated MT system. The problems with MT system can be understood with the help of PE; it automatically tests the MT system as well as evaluates it. Post Editing can be used in evaluating the quality of translation. It can be calculated based on correction or effort required in fixing the issues observed in automated machine translation.

As we can see more accurate translation needs less editing. Translated text from automated MT system is edited by human annotators. Post editing of translated text (output of MT System) is quicker compared to the one where translation is initiated from scratch by a human translator.

This paper discusses the challenges in designing the automated evaluation metrics for English-to-Hindi MT. In Sect. 2 we will talk about the brief history of human and automated evaluation metrics. Section 3 elaborates the automated evaluation. Issues in MT Evaluation with examples of Hindi sentences are discussed in Sect. 4. Section 5 elaborates on statistical analysis for evaluation of such metrics and error classification. Finally we conclude our discussion summarizing the issues in existing automatic English-to-Hindi MT evaluation metrics and provide suggestions for future work in this domain.

2 Related Work

Human evaluation of Machine Translation system is highly subjective and time consuming and it cannot be reused. Due to above mentioned reasons, nowadays automatic evaluation methods are more common. Widely and popularly known algorithm for evaluating the quality of machine translated text is BLEU [3]. It calculates n-gram precision and a brevity penalty between the candidate and reference translation. Some alterations have been done in BLEU metric to come up with NIST metric. NIST metric provides weights to n-gram based on how informative this n-gram is. Lower weight will be given for frequent n-grams. It is proposed by Doddington [4]. Turian et al. [5] introduced GTM (General Text Matcher) based on accuracy measures such as precision, recall and f-measure. The package for automatic evaluation of summaries is introduced in 2003 named as ROUGE [16]. In ROUGE, computer generated summary (candidate summary) is evaluated with the human created summary (reference summary). METEOR [15] came in 2005 that creates a word alignment between the two sentences i.e. candidate translation string and reference translation string. The alignment is done through a word mapping such as i) Stem matching, ii) Exact matching, iii) Synonym matching. After getting the final alignment, score is calculated as the harmonic mean of unigram precision and recall. In recent years extension of METEOR translation evaluation metric is done to the phrase level in METEOR NEXT metric [8]. Additional paraphrase matcher is introduced where phrases and words are matched.

Different human judgements are explored with Human mediated Translation Edit Rate (HTER) [9]. HTER is a semi-automatic measure where humans do not score the MT output but they generate a new reference translation which is closer to MT output and try to retain the fluency and adequacy of the original translated reference. The translated reference translation is used when the evaluation of MT is performed using Translation Edit Rate [9] or with other automated metrics. Annotator tries to minimize the number of edit to get the reference translation. So post editing is also one of the methods of estimating translation quality, more accurate translations require less editing. It is also helpful for finding the errors in the MT output. Correlation between automatic evaluation matrices with human judgements was attempted by Callison in 2007. They have tried to determine which automatic system produces the highest quality of translation from the list of nine different automatic evaluation metrics [2] and ranked them with the help of comprehensive human evaluation. Two categorical scales are currently being used to represent fluency and adequacy of MT system by the human evaluators.

Keeping different features of Hindi in mind, the METEOR Hindi was released by Ankush Gupta et al. [10] which is a modified version of METEOR based on Hindi specific features. METEOR Universal provides language specific evaluation by learning paraphrase table and function word list whereas earlier METEOR required human ranking judgments in the target language. METEOR Universal performed better for Russian and Hindi language [6].

There are various other metrics that were observed during the survey and many other evaluation metrics have released their new versions. During survey, it has been found that there are number of existing metrics but not all the metrics can correlate well with manual evaluation. They cannot work well with all the languages especially with free word order and morphologically rich languages like Hindi.

3 Classification of Hindi Translation Error

Evaluation of Machine Translation can be done more clearly when we are able to find the errors in machine translated outputs. One or more reference translations are required to find the errors in translation. It also helps in comparing the output of MT system with the correct text. Even though it is ambiguous due to several correct translations for the same source sentence, it is a worthwhile exercise to pinpoint the issues with MT and automatic MT evaluation.

A preliminary MT error typology for English language is defined by Llitjós et al. [11] and Vilar has extended the error classification scheme [7].

We have classified the English-to-Hindi Translation errors in three segments. The errors have a hierarchal structure as shown in Fig. 1 below.

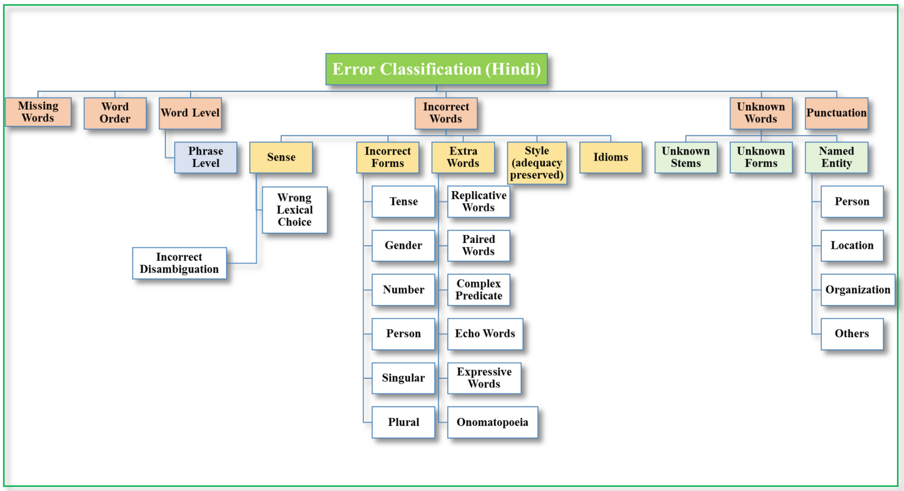


Fig. 1. Hierarchal structure of errors for English-to-Hindi automatic MT

At the first level we have split the errors in eight big classes: “Missing Word”, “Word Order”, “Incorrect Words”, “Extra Words”, “Style”, “Idioms”, “Unknown Words” and “Punctuation”.

A “Missing Word” error is originated when few words are missing in translated sentence. Sometimes in Hindi translation missing of words is mandatory for expressing the translation. Eg: “It is raining outside” the translation in Hindi will be “बाहर बारिश हो रही है। baahar baarish ho rahii hai”. Meaning of “It” is missing in target translation to provide the *naturalness* in the translation.

There are several examples when missing word is essential for expressing the meaning of a source sentence. Missing words can be noun, verb or parsarg. Eg: Ram has two kids. “Raghav and Rajni have two kids” राघव और रजनी के दो बच्चे हैं। “raaghav aur rajanee ke do bachche hain.” When we tried to translate using Google it is giving output with missing “ke” “raaghav aur rajanee do bachche hain.” राघव और रजनी दो बच्चे हैं।. This translation changes the meaning of source sentence due to missing parasarg from the original sentence to “Raghav and Rajni are two kids”.

The second class of error is related to “word order”. Hindi is free word order language but sometimes word order makes a huge difference in the sense of the translated sentence. If we talk about the sentence like क्या आपने खाना खाया ? “kyaa aapne khaana khaaya?” and आपने क्या खाना खाया ? “aapne kyaa khaana khaaya?” Both the sentences have different meaning based on the position of word “kyaa” in the sentence. When “kyaa” is in first position then a person is asking that *Have u taken meal?* But when “kyaa” comes in second position then its sense is that *What items have you taken in meal?*

In the following sentence “not” has important place in target translation. If ‘not’ is associated with banana, then the translation will not be able to capture the correct sense and will change the meaning. “Raghav likes banana not grapes.” Google: राघव नहीं

केला अंगूर पसंद करता है । raaghava nahiin kelaa anguura pasand karta hai. Correct: राघव केला पसंद करता है न की अंगूर। raghav kela pasand karta hai na ki angura.

Errors due to wrong lexical choice and incorrect disambiguation fail to capture the correct sense of the word making the translation inaccurate. In a sentence “I want to meet American head.” “Head” is chief not a body part. We sometimes get Google translation as मैं अमेरकिी सरि से मलिना चाहता हू । The right lexicon choice for the given sentence is”Mukhiya” मैं अमेरकिी मुखयिा से मलिना चाहता हूँ। He had a great fall Google: वह एक महान गरि गया ।Correct: वह धड़ाम से गरि गया । vaha dhadhaam se aa giraa. Example of wrong lexical choice and complex predicate: “The play is on.” Google: खेलने पर है । Correct: खेल चल रहा है । Khela chala rahaa hai.

Incorrect forms of words related to GNP or singular/plural etc. lies in “Incorrect Words” category. Sentences like “A Man scolded a boy.” एक आदमी ने लड़का को डांटा । Eka aadmi ne ladhkA ko dantaa. एक आदमी ने लड़के को डांटा । Eka aadmi ne ladhke ko dantaa.

The additional word normally increased when we observe translation of expressive, replicative, paired or echo words in Hindi.“My hands are sticky”. Google: मेरे हाथ अवरुद्ध कर रहे हैं। Correct: मेरे हाथ चपि चपिा रहे हैं। This is a story of every house. Google: यह हर घर की कहानी है। Naturalness is better in the following sentence: यह घर घर की कहानी है। Please have some snacks. Google: कृपया कुछ नाश्ता लें । More fluent translation is: कृपया कुछ नाश्ता वाश्ता लें । Little fingers are so cute. Google: छोटी उँगलयिों बहुत सुन्दर हैं । Replicative words give more fluent translation: छोटी छोटी उंगलयिां बहुत सुन्दर हैं ।

Interpretation of complex predicate is also an important factor. “He escaped drowning” Google: वह डूबने भाग नकिले । vaha duubte duubte bacha. “वह डूबते डूबते बचा ।” is a correct translation.

4 Automatic Evaluation

In this section we will talk about the methods to calculate the scores in automatic evaluation of MT system. There is also a brief discussion on few automated evaluation metrics.

Normally all automatic metrics of MT evaluation follow one of the following methods:

Precision Based: Total number of matched unigrams between candidate and reference translation are divided by the total length of the candidate translation of MT system.

Recall Based: Total number of matched unigrams between candidate and reference translation are divided by the total length of the reference translation of MT system.

F-measure Based: Collective scores of both precision and recall is being used.

Edit Distance Based: The number of insertion, deletion and substitution is counted for making candidate translation as reference translation.

4.1 BLEU

Papineni proposed BLEU (Bilingual Evaluation Understudy) based on n-gram metric. It evaluates candidate translations produced by an MT system and comparing them with reference translation done by human. For each n (ranges from 1 to maximum of 4) matching is done between candidate and corresponding reference translation. To compensate the difference in the length of candidate and reference translation the brevity penalty is used.

The final BLEU formula is

$$BLEU = BP \times \exp\left(\sum_{n=1}^{4} \frac{1}{n} \log(pn)\right)$$

where pn = modified n gram precision

Brevity penalty is calculated as

$$BP = \min(1.e^{1-lr/lc})$$

where lc = length of the candidate translation

lr = effective reference corpus length.

Discussion Based on Hindi. In this section we will try to compile some examples from Hindi language where BLEU fails to score the MT evaluation**.** We have done both sentence level and corpus (multiple sentences) based analysis for the metrics. The detailed statistical analysis has been provided in Sect. 5. This section explores the issues from a qualitative linguistic perspective (Table 1).

Table 1. Example

E: I am thirsty H: मैं प्यासा हूँ । C1: मुझे प्यास लग रही है ।
E-Source language, H- Human Reference Translation, C- Candidate Translation

BLEU is not able to correlate well with human judgements in all scenarios. In the above example, unigram precision is 0 out of 6; bigram precision is 0 out of 5 and so on. BLEU is not able to capture adequacy of the translation.

Table 2. Example

English Sentence	Human Reference Translation	Candidate Translation
The boy who is standing there is my brother	वो लड़का वहाँ खड़ा है मेरा भाई है ।	लड़का है जो वहाँ खड़ा है मेरा भाई है ।
Even today he breaks down when he reminded of you	तुम्हारी याद करके वह आज भी रो पडता है।	वह आप की याद दलिा दी जब तक कि आज वह टूट जाती है।
Please check where the hotel is	होटल तो देख लो की कहाँ है।	कृपया नरिीक्षण करें जहाँ होटल है ।
Anil used to say that but now his father also says same	अनलि तो कहता ही था अब् उनके पिता भी यही कहते हैं।	अनलि का कहना है कि करते थे. लेकनि अब उनके पिता भी एक ही कहते हैं।
Just look at the watch	जरा घड़ी को तो देख लो ।	आप सरि्फ घड़ी देखयि ।

All the above four examples in Table 2 are revealing diverse features of Hindi language like the role of verb phrase, post positions and adverb. Candidate translations are failing in emphasizing the words whereas human translation could do that with the help of adverbs or postpositions.

After detailed analysis as presented in Sect. 5 and based on above observations, we can conclude that BLEU metric disappoints if the target language is Hindi.

4.2 METEOR

METEOR is an automatic metrics for MT system evaluation based on the concept of unigram matching between the MT systems produced translation and Human reference translation. METEOR is designed after observing the weaknesses of BLEU metrics. It is based on word to word alignment between machine translation and reference translation. Alignment between two sentences can be achieved by exact matching of words if their surface forms are identical. METEOR also matches words with simple morphological variants that can be aligned. If stems are identical and have similar synonym sets then matching will be done between system generated translation and reference translation.

The score is calculated as harmonic mean of unigram precision (matched n-grams out of the total number of n-grams in a MT system produced translation) and unigram recall (matched n-grams out of the total number of n-grams in reference translation). In METEOR-NEXT [8], paraphrase matcher is being introduced. It matches the phrases between the two strings.

Table 3. Example

E: Raghav had beaten Radha with stick H: राघव ने राधा को डंडे से मारा। C1: राधा को राघव ने डंडे से मारा । C2: राधा ने राघव को डंडे से मारा ।
E-Source language, H- Human Reference Translation, C- Candidate Translation

METEOR-Hindi [10] used word based features and other linguistic parameters such as local word groups, part of speech tags and clause boundaries. Local Words Groups (LWG) [13] is a group of content and its associated function words. Function words tell the grammatical role of the content word in the sentence. Example sentence from Table 3 is showing the importance of Local Words Group. If we observe C1 and C2, we will see that all the words in both the sentences are showing exact match with the human reference sentence but C1 and C2 have opposite meaning. With the help of LWG, we can look into the difference and can assign the scores accordingly.

If all the words of a sentence are matched in METEOR-Hindi but POS is not same for the words then the sentence will get low scores. METEOR-Hindi also computes the exact matching of clauses and gives the scores according to the matched clauses.

4.3 TER (Translate Error Rate)

TER measure is proposed by Dorr and Snover in 2006 [14]. It calculates the amount of editing required in a MT system output to achieve the exact match of reference translation. TER counts the number of edits based on fluency and adequacy of a sentence. In TER we do not generate a new reference but try to match the system output with existing references.

$$TER = \frac{number\ of\ edits}{average\ of\ reference\ words}$$

In HTER (Human Mediated Targeted Error Rate), we find minimum edits required as per the new targeted reference generated by human. HTER is more complex as a human does not score directly on the MT output. Instead they generate a new reference translation which is closer to the MT output translation.

TER_p is an extension of TER where alignment is not based on exact match; rather it takes synonyms and stem of a word while matching. It does Stem match, Synonym match and phrase substitutions (Table 4).

Table 4. Example

E: Father scolded one boy H: पिता ने एक लड़के को डांटा । C: पिता ने एक लड़का को डांटा ।
E-Source language, H- Human Reference Translation, C- Candidate Translation

We can observe how post editing can improve the accuracy of translation. We can also consider replicative words in evaluation of machine translation. Replicative words can occur in almost all the South Asian Languages [15]. Replicative words enhance the naturalness of the translation. They improve the fluency of the translated output (Table 5).

Table 5. Example

E: This is a story of every house H: यह घर घर की कहानी है। C: यह प्रत्येक घर की कहानी है ।
E-Source language, H- Human Reference Translation, C- Candidate Translation

5 Evaluation and Error Analysis

Based on qualitative insights being gathered, as described in previous section, a detailed statistical analysis has been performed to quantify the correlations for BLEU and METEOR against human evaluation.

5.1 Sample Input Data

148 small paragraphs were randomly selected from following sources:

- 92 Paragraphs from Online Course Material from National Institute of Open Schooling (NIOS) [17]
- 32 Paragraphs from Online Stories
- 24 Paragraphs from Government Websites [18]

NIOS provides educational material for vocational, secondary and senior secondary courses in both English and Hindi languages. Course content is developed in English language. Hindi content for corresponding course material is then generated using the services of human translators. These are considered as reference translations whereas machine translations have been obtained using Google translator [19].

Second set of data has been collected using online Indian stories in English language. Examples include stories from Premchand, Akbar-Birbal anecdotes etc. 32 randomly selected paragraphs from these sources were given to people with Hindi as their native language and who can use English fluently as second language. Again these were translated using Google translate as well for further evaluation.

India has 22 official languages including Hindi and English. Government websites are generally multilingual, so we have taken 24 paragraphs from there in both English and Hindi languages.

5.2 Error Classification

From the data generated, 32 paragraphs were randomly selected for determining the top 5 Class of Errors. The distribution of errors for these randomly selected statements is shown in Table 6 below.

Table 6. Errors distribution

Class of errors	Count
Missing Word	40
Wrong Lexicon Choice	72
Extra Word	40
Word Order	66
Incorrect Forms	20

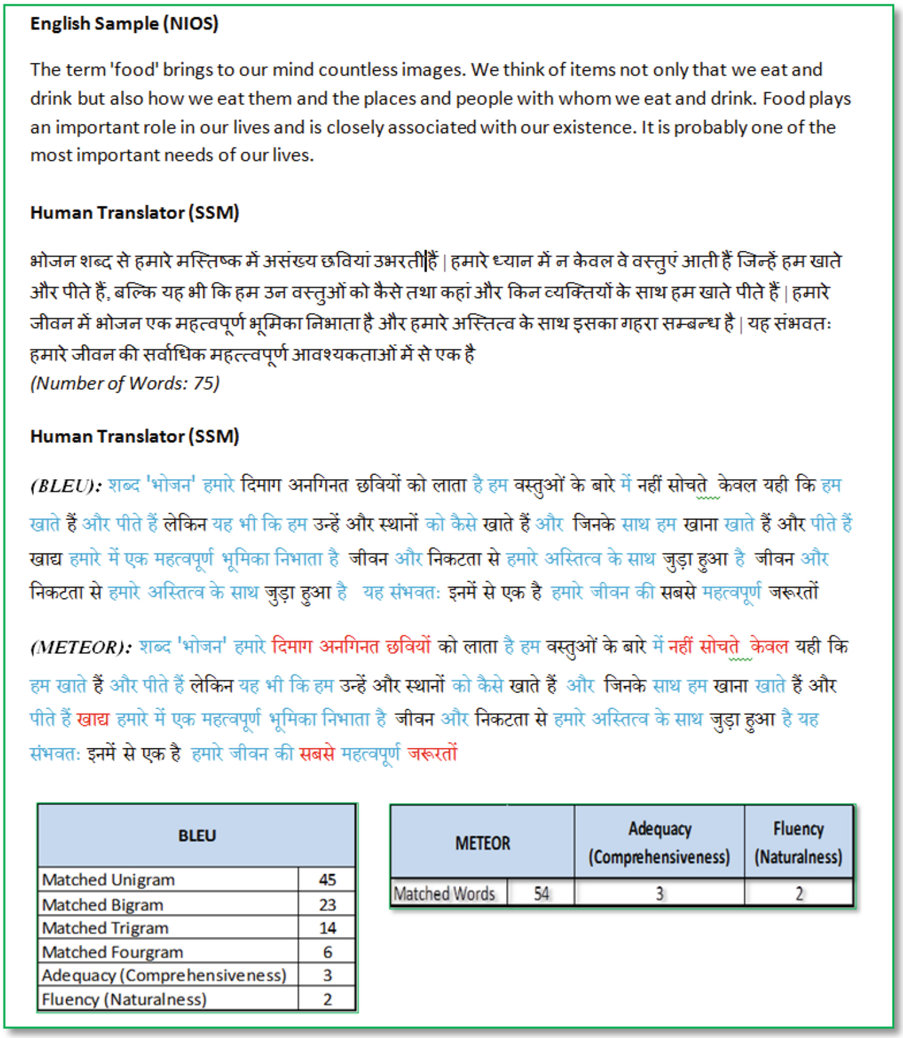
5.3 Metric Evaluation

BLEU and METEOR are selected for English-to-Hindi MT metric evaluation. All 148 statements have been analysed. Python NLP library (NLTK [20]) was used to tokenize the sample paragraphs. Subsequently, a Python program was written to compare the two translations (Human vs Automatic) for matched unigrams, bigrams, trigrams and four-grams. 32 of these 148 statements were also analysed manually. Figure 2 shows the sample of task carried out manually, whereas, Table 7 below shows the recorded observations.

Following formulae were used to evaluate BLUE and METEOR scores. Precision, Recall and F-Measure have been obtained using standard formulae as described in Sect. 4. Table 8 compiles all the results thus obtained.

$$BLEU\,Score = MIN\left(1, \frac{output\,length}{reference\,length}\right) \times \left(\prod_{i=1}^{4} precision_i\right)^{\frac{1}{4}}$$

$$METEOR\,Score = \frac{10PR}{R + 9P}$$



English Sample (NIOS)

The term 'food' brings to our mind countless images. We think of items not only that we eat and drink but also how we eat them and the places and people with whom we eat and drink. Food plays an important role in our lives and is closely associated with our existence. It is probably one of the most important needs of our lives.

Human Translator (SSM)

भोजन शब्द से हमारे मस्तिष्क में असंख्य छवियां उभरती|हैं | हमारे ध्यान में न केवल वे वस्तुएं आती हैं जिन्हें हम खाते और पीते हैं, बल्कि यह भी कि हम उन वस्तुओं को कैसे तथा कहां और किन व्यक्तियों के साथ हम खाते पीते हैं | हमारे जीवन में भोजन एक महत्वपूर्ण भूमिका निभाता है और हमारे अस्तित्व के साथ इसका गहरा सम्बन्ध है | यह संभवतः हमारे जीवन की सर्वाधिक महत्त्वपूर्ण आवश्यकताओं में से एक है
(Number of Words: 75)

Human Translator (SSM)

(BLEU): शब्द 'भोजन' हमारे दिमाग अनगिनत छवियों को लाता है हम वस्तुओं के बारे में नहीं सोचते केवल यही कि हम खाते हैं और पीते हैं लेकिन यह भी कि हम उन्हें और स्थानों को कैसे खाते हैं और जिनके साथ हम खाना खाते हैं और पीते हैं खाद्य हमारे में एक महत्वपूर्ण भूमिका निभाता है जीवन और निकटता से हमारे अस्तित्व के साथ जुड़ा हुआ है जीवन और निकटता से हमारे अस्तित्व के साथ जुड़ा हुआ है यह संभवतः इनमें से एक है हमारे जीवन की सबसे महत्वपूर्ण जरूरतों

(METEOR): शब्द 'भोजन' हमारे दिमाग अनगिनत छवियों को लाता है हम वस्तुओं के बारे में नहीं सोचते केवल यही कि हम खाते हैं और पीते हैं लेकिन यह भी कि हम उन्हें और स्थानों को कैसे खाते हैं और जिनके साथ हम खाना खाते हैं और पीते हैं खाद्य हमारे में एक महत्वपूर्ण भूमिका निभाता है जीवन और निकटता से हमारे अस्तित्व के साथ जुड़ा हुआ है यह संभवतः इनमें से एक है हमारे जीवन की सबसे महत्वपूर्ण जरूरतों

BLEU	
Matched Unigram	45
Matched Bigram	23
Matched Trigram	14
Matched Fourgram	6
Adequacy (Comprehensiveness)	3
Fluency (Naturalness)	2

METEOR		Adequacy (Comprehensiveness)	Fluency (Naturalness)
Matched Words	54	3	2

Fig. 2. Sample of Evaluations for English-to-Hindi MT

Table 7. Observations

S. No.	Paragraph #	Exact Matched Words-Unigrams BLEU	Total Words in Reference Translation	Total Words in Candidate Translation	Matched Words-Stem, Synset, Paraphrase-METEOR
1	Para 1	45	75	78	54
2	Para 2	26	66	56	44
3	Para 3	71	182	159	110
4	Para 4	48	109	107	71
.	.	.	.	.	.
.	.	.	.	.	.
147	Para 147	52	100	96	70
148	Para 148	76	194	169	118

Table 8. Metric evaluation

S. No.	Paragraph #	Precision (Unigram)	Precision (Bigram)	Precision (Trigram)	Precision (Fourgram)	Recall	F-Measure	BLEU (Only Unigram)	BLEU (Till Bigram)	BLEU (Till Fourgram)	Meteor Hindi
1	Para 1	0.5769	0.2987	0.1842	0.0800	0.6000	0.5882	0.5769	0.4151	0.2245	0.5976
2	Para 2	0.4643	0.2545	0.1296	0.0755	0.3939	0.4262	0.3939	0.2917	0.1565	0.4000
3	Para 3	0.4465	0.2595	0.1592	0.1026	0.3901	0.4164	0.3901	0.2974	0.1822	0.3951
4	Para 4	0.4486	0.2830	0.1810	0.1154	0.4404	0.4444	0.4404	0.3498	0.2227	0.4412
.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.
147	Para 147	0.5412	0.3157	0.2021	0.1290	0.5202	0.5305	0.5202	0.3973	0.2483	0.5222
148	Para 148	0.4485	0.1790	0.1141	0.0725	0.3906	0.4175	0.3906	0.2467	0.1398	0.3957

5.4 Adequacy (Comprehensiveness) and Fluency (Naturalness) Evaluation

BLEU and METEOR scores for 32 randomly selected paragraphs have been compared against the adequacy and fluency scores for English-to-Hindi MT provided by human evaluator on the same set of 32 paragraphs. Adequacy and Fluency scores are categorical with the scales shown in Table 9 below.

Table 9. Adequacy and fluency scales

Adequacy (Comprehensiveness)		Fluency (Naturalness)
All Meaning (5)		Flawless Hindi (5)
Most Meaning (4)		Good Hindi (4)
Much Meaning (3)		Non-Native Hindi (3)
Little Meaning (2)		Disfluent Hindi (2)
None (1)		Incomprehensible (1)

Pearson Correlation Coefficient has been obtained to understand the behaviour of BLEU and METEOR with respect to adequacy and fluency. Figure 3 provides the graphical interpretation of this correlation for BLEU whereas Fig. 4 provides the same for METEOR scores.

Pearson Correlation Coefficient is given by:

$$r_{xy} = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{(n-1)s_x s_y}$$

where, $\bar{x}, \bar{y}$ are mean values and s_x, s_y are square roots of variance.

As can be clearly seen from the graphed data, METEOR performs much better in terms of both adequacy and fluency compared to BLEU. METEOR gives a correlation coefficient of more than 0.75 for both adequacy and fluency whereas BLEU has a correlation coefficient of 0.51 for adequacy and 0.61 for fluency.

The correlation for BLEU is especially bad whereas METEOR gives a correlation which is somewhat acceptable with lot of room left for further improvements.

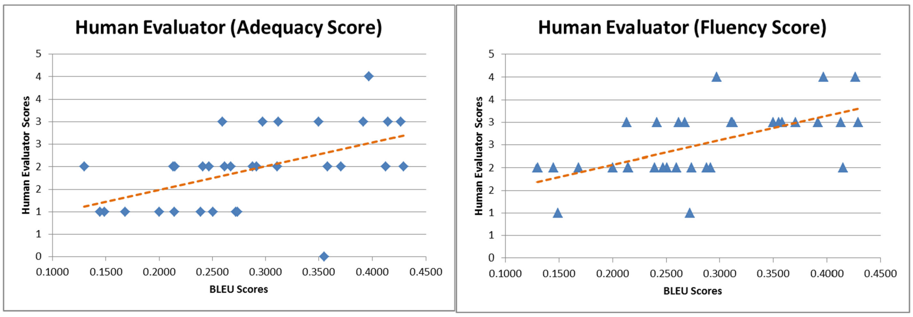


Fig. 3. BLEU vs. human evaluator

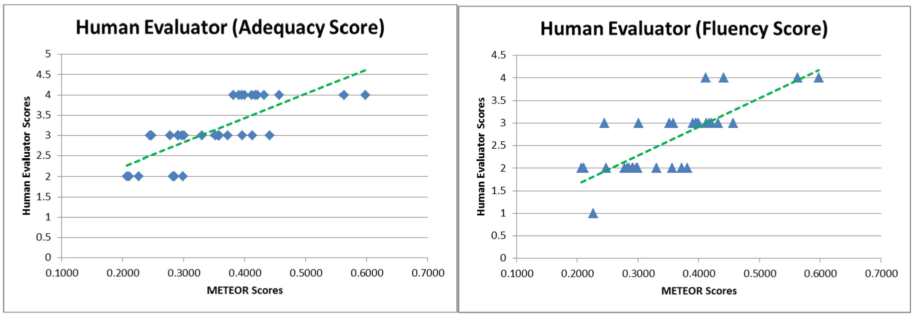


Fig. 4. METEOR vs. human evaluator

6 Conclusion

In the paper we discussed various important features of Hindi language and the anomalies that are encountered during evaluation of available translation systems. Various metrics have been studied and it has been established that these metrics are not able to evaluate English-to-Hindi translation appropriately.

We found that Post Position plays an important semantic role in Hindi. We should measure the Post Position Equivalence information to add one more level of matching word group. Even though Hindi is relatively free word order but Wrong Word/Phrase order impacts the naturalness of the MT translation. Incorrect form of verb influences the comprehensiveness of the text. We need to identify word groups such as NN + PSP, ADJ + NN and Verb groups, which includes verb and auxiliary verbs. We can score evaluation metrics based on strong and weak equivalence and Head Word matching of the reference and candidate translation.

As guidance for future work, all these parameters along with the tagged data obtained during the analysis contained in this paper can be utilized as input features for machine learning algorithms including neural nets to improve the performance of automatic English-to-Hindi MT evaluation metrics.

References

1. Specia, L., et al.: Predicting machine translation adequacy. Machine Translation Summit XIII, Xiamen, China (2011)
2. Callison-Burch, C., et al.: (Meta-) evaluation of machine translation. In: Proceedings of the Second Workshop on Statistical Machine Translation. Association for Computational Linguistics (2007)
3. Papineni, K., et al.: BLEU: a method for automatic evaluation of machine translation. In: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics. Association for Computational Linguistics (2002)
4. Doddington, G.: Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In: Proceedings of the Second International Conference on Human Language Technology Research. Morgan Kaufmann Publishers Inc. (2002)
5. Turian, J.P., Shen, L., Melamed, I.D.: Evaluation of machine translation and its evaluation. In: Proceedings of MT Summit IX, New Orleans, U.S.A (2003)
6. Denkowski, M., et al.: Meteor universal: Llanguage specific translation evaluation for any target language. In: Proceedings of the Ninth Workshop on Statistical Machine Translation, Baltimore, Maryland USA, June 26–27, 2014, pp. 376–380. Association for Computational Linguistics (2014)
7. Vilar, D., et al.: Error analysis of statistical machine translation output. In: Proceedings of LREC (2006)
8. Denkowski, M., Lavie, A.: Extending the METEOR machine translation evaluation metric to the phrase level. In: Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics. Association for Computational Linguistics (2010)
9. Snover, M., et al.: A study of translation edit rate with targeted human annotation. In: Proceedings of Association for Machine Translation in the Americas (2006)

10. Gupta, A., Venkatapathy, S., Sangal, R.: METEOR-Hindi: automatic MT evaluation metric for Hindi as a target language. In: Proceedings of ICON-2010: 8th International Conference on Natural Language Processing (2010)
11. Llitjós, A.F., Carbonell, J.G., Lavie, A.: A framework for interactive and automatic refinement of transfer-based machine translation. In: Proceedings of the 10th Annual Conference of the European Association for Machine Translation (EAMT), Budapest, Hungary (2005)
12. Kalyani, A., et al.: Assessing the Quality of MT Systems for Hindi to English Translation (2014). arXiv preprint arXiv:1404.3992
13. Bharati, A., Chaitanya, V., Sangal, R.: Local word grouping and its relevance to Indian languages. In: Bhatkar, V.P., Rege, K.M. (eds.) Frontiers in Knowledge Based Computing (KBCS90), pp. 277–296. Narosa Publishing House, New Delhi (1991)
14. Ananthakrishnan, R., et al.: Some issues in automatic evaluation of English-Hindi mt: more blues for bleu. ICON **64** (2007)
15. Banerjee, S., Lavie, A.: METEOR: an automatic metric for MT evaluation with improved correlation with human judgments. In: Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization (2005)
16. Hovy, E., et al.: Automated summarization evaluation with basic elements. In: Proceedings of the Fifth Conference on Language Resources and Evaluation (LREC 2006)
17. NIOS Online Course Material. http://www.nios.ac.in/online-course-material.aspx
18. Center for Development of Advanced Computing, Govt. of India. http://www.cdac.ac.in
19. Google Translate. https://www.translate.google.com
20. NLTK NLP Library in Python. http://www.nltk.org

Morphology, Syntax

Unsupervised Learning of Word Segmentation: Does Tone Matter?

Pierre Godard[1], Kevin Löser[1], Alexandre Allauzen[1], Laurent Besacier[2], and François Yvon[1](✉)

[1] LIMSI, CNRS, Université Paris-Saclay, Orsay, France
{godard,loser,allauzen,yvon}@limsi.fr
[2] Laboratoire d'Informatique de Grenoble (LIG), Université Grenoble Alpes, Grenoble, France
laurent.besacier@imag.fr

Abstract. In this paper, we investigate the usefulness of tonal features for unsupervised word discovery, taking Mboshi, a low-resource tonal language from the Bantu family, as our main target language. In a preliminary step, we show that tone annotation improves the performance of *supervised learning* when using a simplified representation of the data. To leverage this information in an unsupervised setting, we then present a probabilistic model based on a hierarchical Pitman-Yor process that incorporates tonal representations in its backoff structure. We compare our model with a tone-agnostic baseline and analyze if and how tone helps unsupervised segmentation on our small dataset.

Keywords: Tonal features · Mboshi · Hierarchical Pitman-Yor process

1 Introduction

Many languages will face extinction in the coming decades. Half of the 7,000 languages spoken worldwide are expected to disappear by the end of this century [2], and there are too few field linguists to document all of these endangered languages. Innovative speech data collection methodologies [4,5] as well as computational assistance [1,15] were recently proposed to help them in their documentation and description work. This paper follows similar objectives and focuses on the unsupervised discovery of words from an unsegmented sequence of symbols. While such material could be obtained by automatic phone recognition from speech, we investigate here oracle data (hand-annotated symbols from linguists) in a rather limited resource setting (only a few thousand sentences). In this context, our main question is to evaluate the usefulness of tone information in unsupervised word segmentation, and we use Mboshi, a mostly unwritten African language of the Bantu family, as our main test case.

The integration of tonal information in segmentation relies on Bayesian non-parametric (BNP) models, popularized in Natural Language Processing (NLP) by [8–10]. In this approach, (pseudo)-words or (pseudo)-morphs are generated by a bigram model over a non-finite inventory, through the use of a Dirichlet process (DP), or a more general Pitman-Yor process (PYP), which enables to discover units that follow a power-law distribution, a universal characteristic of language lexicons. These algorithms were originally designed as computational models of language acquisition and were mainly applied to non-tonal languages, with the exception of [11], who investigated the use of adaptor grammars for unsupervised word segmentation of Mandarin Chinese. Tones were shown to have a small impact on segmentation accuracy and were reported to yield a small improvement for simple grammars and no improvement with more complex ones. Also worth mentioning is the work of [12], who studied the role of prosodic information (at a macroscopic level) for word segmentation. The approach was tested on English and Japanese: for both languages, it was shown that prosodic boundaries were actually helping word segmentation.

Contributions: we investigate in this paper whether and how tone annotation can help word segmentation in the case where the distribution of tones obey morphological and syntactical, as well as lexical, constraints. After briefly presenting some peculiarities of Mboshi, we first show (in Sect. 2) that tones help disambiguate word boundaries in a *supervised setting* - suggesting that this information could also help the unsupervised discovery of words. We then present in Sect. 3 a new hierarchical BNP model, which uses generalized backoff schemes to integrate tonal representations in word segmentation. Based on the experiments reported in Sect. 4, we conclude that, notwithstanding the inherent unstability of BNP models, tonal information can improve word discovery procedures.

2 A Preliminary Study: Supervised Word Segmentation

In order to assess the potential for tones to help discovering a tonal language's word units, we first conducted a supervised experiment making use of decision trees. We start this section by presenting a short sketch of the language targeted in this work: Mboshi.

2.1 Mboshi Language

Mboshi is a language spoken in Congo-Brazzavile, and it was one of the languages documented by the BULB (Breaking the Unwritten Language Barrier) project [1, 15], using the LIG-AIKUMA tool [5]. Preliminary experiments for a small portion of it were reported in [7]. Mboshi is a two tone Bantu language whose words are typically composed of roots and affixes. Almost all Mboshi words include at least one prefix, while the presence of several prefixes and one suffix is also very common. While the language can be considered as rarely written, linguists have nonetheless defined a non-standard grapheme form of it, considered to be close to the language phonology. The suffix structure tends to be a single vowel (e.g. -a or -i) whereas the prefix structure may be both CV and V. Most common syllable structures are V and CV, although CCV may arise due to affricates and pre-nasalized plosives (coded with symbols *dz* and *mb*). Mboshi also makes use of short and long vowels (coded respectively as V and VV). With respect to tones, the high tone is coded using an acute accent on the vowel while low tone vowel has no special marker. Word root, prefix and suffix all bear specific tones which tend to be realized as such in their surface forms.[1] Tonal modifications may also arise from vowel deletion at word boundaries. Concerning grammatical tones, word root, prefix and suffix all bear specific tones which tend to be realized as such in their surface forms. Tonal modifications may arise from vowel deletion at word boundaries. A productive combination of tonal contours in words can also take place due to the preceding and appended affixes. These tone combinations play an important grammatical role particularly in the differentiation of tenses. However, in Mboshi, the tones of the roots are not modified due to conjugations, unlike in many other Bantu languages.

2.2 Data and Representations

Our study uses a corpus in Mboshi built both from translated reference sentences for oral language documentation [6] and from a Mboshi dictionary [3]. This corpus, comprising more than 9K sentences, is split in three parts that we call `S`, `M` and `L` and for which we give basic statistics in Table 1. Subset `S` can be considered as homogeneous, as it comes from a single source; subsets `M` and `L` come from two different sources and exhibit more lexical diversity.

Table 1. Corpus statistics for the Mboshi corpus (`letter+tone` representation).

Name	#sent	#tokens	#types
`S`	1,174	6,238	1,664
`M`	4,904	27,990	7,271
`L`	9,256	52,433	11,440

In our transcriptions, Mboshi's tonal system consisting of a pair of high and low tones is simply represented using diacritics on vowels: acute accent for a high tone, and no accent for a low tone. Our approach consists in varying the representation of the input text and comparing the full transcription with diacritics (`letter+tone`) to i) the transcription `letter` where diacritics are removed, ii) the transcription `xV` where vowels are replaced by the symbol 'V', iii) the transcription `xLH` where high-toned vowels are replaced by the symbol 'H' and low-toned vowels are replaced by 'L', iv) the transcriptions `CV` (resp. `CLH`) where consonants in `xV` (resp. `xLH`) are replaced by a generic symbol, 'C' (see Table 2). We expect the systematic comparison of tonal (`letter+tone`, `xLH`, and `CLH`) with their non tonal counterpart (`letter`, `xV`, `CV`) to shed some light on the usefulness of this information.

[1] The distinction between high and low tones is phonological (see [14]).

Table 2. Various representations of the text.

Name	Representation
`letter`	wa ay *ee* la midi
`letter+tone`	wa áy*ee* la midí
`CV`	CV VCVV CV CVCV
`CLH`	CL HCLL CL CLCH
`xV`	wV VyVV lV mVdV
`xLH`	wL HyLL lL mLdH

2.3 Disambiguating Word Boundaries with Decision Trees

For each representation of the text, we train a decision tree classifier[2] to predict a binary decision corresponding to the presence or absence of a word boundary after each character. This prediction is based on features encoding a fixed-length window of characters centered around the decision point; and consider in our experiments windows of varying size[3] We report the precision, recall and F-measure computed on ambiguous[4] word boundaries in Table 3 on the `S` corpus[5] where 10% of the number of sentences have been held out for testing purposes. For the pseudo-orthographic (`letter` and `letter+tone`) text representations, it seems that the tonal information is of little value to disambiguate the frontiers. However, as we simplify the text representation, despite a drop in F-measure, the contrast between a representation ignoring tones (`CV`) and one that captures them (`CLH`) becomes much sharper, suggesting that a tonal signal can be used to improve segmentation. This motivates the design of new models that could capture this signal directly on pseudo-orthographic representations and help improve the precision of unsupervised segmentation.

Table 3. Precision, Recall and F-measure on word boundaries in various text representations of corpus `S` with a decision tree classifier (11-words window width).

Representation	P	R	F-measure
`letter`	0.92	0.93	0.92
`letter+tone`	0.91	0.89	0.90
`xV`	0.86	0.89	0.87
`xLH`	0.88	0.89	0.88
`CV`	0.70	0.61	0.65
`CLH`	0.78	0.72	0.75

[2] We use `scikit-learn`'s implementation (http://scikit-learn.org/stable/modules/tree.html).

[3] With padding at the beginning and end of the sentence.

[4] We exclude word boundaries corresponding to the beginning and end of the sentence.

[5] Similar results are obtained for the larger corpora.

3 Non-parametric Segmentation Models with Tone Information

3.1 Pitman-Yor Processes

PYPs are a class of stochastic processes used as models for sparse probability distributions with a countably infinite support and distributed according to a power-law, and therefore especially suited to model distributions arising in linguistic data [9,16]. A PYP is defined by some *base distribution* P_0, and two *concentration* ($\theta \in]-d, \infty[$) and *discount* ($d \in [0, 1[$) hyperparameters, and generates sparse versions of P_0, whose degree of sparsity is controlled by θ and d. Rather than explicitly defining $\text{PYP}(P_0, \theta, d)$, we use the Chinese Restaurant Process (CRP) metaphor to recall its main properties.

We assume a restaurant with K tables, with n_k customers seated at table k ($k \in \{1, \ldots, K\}$), and $N = \sum_{k=1}^{K} n_k$ the total number of customers. Each table has a label l_k from the domain $\mathcal{D}(P_0)$ of P_0. The restaurant is initially empty, with no table. Customers enter one by one and:

– seat at table k table with probability proportional to $n_k - d$
– seat at an empty table with probability proportional to $\theta + K \cdot d$. l_{K+1} is then chosen by sampling from P_0 and K is incremented.

The table layout defines a distribution $P(l)$ over $\mathcal{D}(P_0)$, where $P(l)$ is the probability of obtaining l by uniformly picking a random customer and returning its table label. When $N \to \infty$, $P(l)$ converges to a sample of $\text{PYP}(P_0, \theta, d)$.

Given a particular table setup, the probability of sampling a label x is computed as:

$$P(x|n_1 \ldots n_K, l_1 \ldots l_K) \propto \left(\sum_{k=1}^{K} \mathbf{1}_{l_k = x} \cdot (n_k - d) \right) + (\theta + Kd) \cdot P_0(x).$$

3.2 Sampling Segmentations Using a PYP-Distributed Unigram Word Model

In this work, we model a sentence as a concatenation of words drawn from a unigram distribution P_w generated by a $\text{PYP}(P_{spl}, \theta, d)$ as in [10]. P_{spl} is the *spelling model*, for instance a n-gram model over character sequences (see below). We tokenize a corpus $s_1 \ldots s_n$ of n sentences by Gibbs-sampling every segmentation s_i conditioned on all other segmentations.[6] As explained above, sampling $x \sim P_w$ can be done by maintaining a CRP associated to P_w, such that for every token t in the current lexicon L there is a customer seated at a table labelled

[6] Following [13], we use a forward filtering-backward sampling (FFBS) algorithm to sample segmentations. As this method only approximates the posterior distribution, we also perform a Metropolis-Hastings correction step.

with the type of t. We also placed agnostic priors on the PYP hyperparameters: $\theta \sim \exp(1)$ and $d \sim Beta(1,5)$ and resampled these hyperparameters as well as the CRP table layouts every 200 iterations.

3.3 A Spelling Model with Tones

The spelling model defines a distribution over character strings that reflects how word forms should look like. Obvious candidate spelling models are n-gram models of character sequences:[7]

$$P_{spl}(w) = \prod_{i=1}^{l} P(c_i|c_{i-n+1}\dots c_{i-1}) \\ \cdot P(\text{stop}|c_{l-n+1}\dots c_l)$$

In order to also integrate tone information in the spelling model P_{spl}, we define the set of *contexts* $\mathcal{K}$ as the set of sequences $\tau_1 \dots \tau_j\, c_{j+1} \dots c_k$ (with $k \in \{0,\dots,n-1\}$), where τ_i are *tones* symbols from the set {H(high tone), L(low tone), C(consonant)} and c_i are regular characters. A *context* is thus a sequence of length at most $n-1$ comprising a prefix of tones and a suffix of characters.

For every non-empty context $\kappa \in \mathcal{K}$, we further assume that the conditional distribution $P(c|\kappa)$ recursively arises from a PYP with base distribution $P(c|\beta(\kappa))$, where β is a *backoff function.* The base distribution of the unigram distribution (κ is empty) is the uniform distribution. In this setting, base distributions of PYPs themselves arise from PYPs, giving rise to a *hierarchical PYP* whose structure is defined by the backoff function β. To enrich the model with awareness of tone patterns, we designed the following backoff scheme, where characters are first replaced by tones (rightwards), then dropped (rightwards), as follows:

$$c_1 \dots c_{n-1} \xrightarrow{\beta} \tau_1 c_2 \dots c_{n-1} \xrightarrow{\beta} \tau_1\tau_2 c_3 \dots c_{n-1} \\ \xrightarrow{\beta} \dots \xrightarrow{\beta} \tau_1 \dots \tau_{n-1} \xrightarrow{\beta} \tau_2 \dots \tau_{n-1} \\ \xrightarrow{\beta} \tau_3 \dots \tau_{n-1} \xrightarrow{\beta} \tau_{n-1} \xrightarrow{\beta} \emptyset.$$

On a Mboshi example, backoff will thus unfold as follows: `ámid` $\xrightarrow{\beta}$ H`mid` $\xrightarrow{\beta}$ HC`id` $\xrightarrow{\beta}$ HCL`d` $\xrightarrow{\beta}$ HCLC $\xrightarrow{\beta}$ CLC $\xrightarrow{\beta}$ LC $\xrightarrow{\beta}$ C $\xrightarrow{\beta}$ $\emptyset$. This model is referred to as MULTI in our experiments. We also evaluated an alternative backoff scheme β', where only one single tone is remembered:

$$c_1 \dots c_{n-1} \xrightarrow{\beta'} \tau_1 c_2 \dots c_{n-1} \xrightarrow{\beta'} \tau_2 c_3 \dots c_{n-1} \\ \xrightarrow{\beta'} \dots \xrightarrow{\beta'} \tau_{n-2} c_{n-1} \xrightarrow{\beta'} \tau_{n-1} \xrightarrow{\beta'} \emptyset$$

[7] where we add initial padding symbols as needed.

This is illustrated on the same Mboshi example: `ámid` $\xrightarrow{\beta'}$ `Hmid` $\xrightarrow{\beta'}$ `Cid` $\xrightarrow{\beta'}$ `Ld` $\xrightarrow{\beta'}$ C $\xrightarrow{\beta'}$ $\emptyset$. This model is referred to as LAST in our experiments.

We compare our tone models MULTI and LAST to a baseline PYP n-gram spelling model—referred to later on in our experiments as BASE—that is unable to distinguish between high and low tones. In this baseline, the backoff scheme β_{BASE} is simply defined by:

$$c_1 \dots c_{n-1} \xrightarrow{\beta_{\text{BASE}}} c_2 \dots c_{n-1} \xrightarrow{\beta_{\text{BASE}}} c_3 \dots c_{n-1}$$
$$\xrightarrow{\beta_{\text{BASE}}} \dots \xrightarrow{\beta_{\text{BASE}}} c_{n-2}c_{n-1} \xrightarrow{\beta_{\text{BASE}}} c_{n-1} \xrightarrow{\beta_{\text{BASE}}} \emptyset$$

It corresponds on the previously used example to: `ámid` $\xrightarrow{\beta_{\text{BASE}}}$ `mid` $\xrightarrow{\beta_{\text{BASE}}}$ `id` $\xrightarrow{\beta_{\text{BASE}}}$ `d` $\xrightarrow{\beta_{\text{BASE}}}$ $\emptyset$.

4 Experiments

Several setups are considered with varying corpus sizes (`S`, `M`, and `L`) and text representations (`letter+tone`, `letter`, etc.). We report precision, recall and F-measure on word boundaries (BP, BR, BF) and word types (LP, LR, LF) with respect to `S` used as a test set for all corpora sizes. Additionally, we make the spelling model's Markov order vary between 1 and 6 and also conduct experiments with the BASE model on all the text representations discussed in Sect. 2.2. Each configuration (144 in total) is ran 5 times resulting in a total of 720 measures. Table 4 gives the results of the best thirty runs in terms of F-measure on word boundaries (BF).

Unlike what was observed in the supervised setting (Sect. 2), there seems to be some benefit in keeping the tonal information in the representation (`letter+tone` vs. `letter`): 22 out of the 30 best runs use tone information and top 9 configurations actually use the `letter+tone` representation. We can also observe that large values of n (beyond 3) do not help much and that $n = 3$ seems to be a good compromise. The benefit of our tone models (MULTI and LAST) is less conclusive when compared to the BASE model, which also obtained very good performance.

Also note that we did not observe a clear trend when increasing the corpus size. This might be due to the more heterogeneous nature of our Mboshi `M`, and `L` subsets mentioned in Sect. 2.2. Of particular interest are the results obtained with the `xLH` representation and the BASE model: notwithstanding the replacement of 14 symbols (Mboshi has a 7 vowels inventory, each prone to carry a low or high tone) by only 2 symbols encoding the tonal information, the performance compares to the very best result on our segmentation task.

To serve as another baseline, we also ran `dpseg` [8, 10][8] for all the possible representations and corpus sizes (18 configurations). It implements a Nonparametric Bayesian approach, where (pseudo)-words are generated by a bigram model over

[8] http://homepages.inf.ed.ac.uk/sgwater/resources.html.

Table 4. Top 30 best F-measures on word boundaries (BF) for all models, corpora sizes, spelling model's Markov order, text representations and 5 runs for each (144 configurations * 5 runs were evaluated in total). A `dpseg` baseline was also run for all possible representations and corpus sizes (18 configurations) and best configuration lead to BF equals to 70.40%.

Corpus size	Model	Markov order	Representation	BP	BR	BF	LP	LR	LF
`M`	BASE	3	`letter+tone`	89.07	72.89	80.17	58.28	62.80	60.46
`S`	LAST	1	`letter+tone`	77.59	81.50	79.50	54.13	45.67	49.54
`M`	LAST	3	`letter+tone`	87.20	72.93	79.43	56.53	60.64	58.51
`M`	BASE	3	`letter+tone`	86.53	71.52	78.31	55.93	60.34	58.05
`L`	BASE	3	`letter+tone`	89.17	69.23	77.95	55.80	62.44	58.93
`S`	MULTI	3	`letter+tone`	88.58	69.08	77.62	52.12	54.03	53.05
`M`	LAST	3	`letter+tone`	83.82	71.48	77.16	52.94	57.87	55.30
`M`	LAST	6	`letter+tone`	76.29	78.04	77.16	47.98	46.45	47.21
`S`	LAST	1	`letter+tone`	76.86	77.47	77.16	54.45	47.42	50.69
`L`	BASE	2	`letter`	81.51	73.20	77.13	55.25	55.21	55.23
`S`	LAST	6	`letter+tone`	80.25	74.23	77.12	53.16	48.56	50.75
`M`	BASE	3	`letter`	88.85	68.13	77.12	57.36	63.46	60.26
`M`	BASE	3	`letter`	87.71	68.33	76.81	57.44	63.12	60.15
`M`	LAST	5	`letter+tone`	84.25	70.56	76.80	50.75	57.09	53.73
`M`	BASE	3	`letter`	89.35	67.24	76.73	56.64	63.39	59.83
`M`	LAST	3	`letter+tone`	85.66	69.23	76.58	52.92	58.29	55.48
`M`	BASE	4	`letter`	94.24	64.32	76.46	54.49	65.37	59.44
`L`	BASE	3	`letter`	89.10	66.96	76.46	54.05	62.78	58.09
`L`	BASE	4	`letter+tone`	90.90	65.88	76.39	52.28	62.08	56.76
`M`	BASE	3	`xLH`	87.96	67.26	76.23	52.88	62.44	57.26
`S`	LAST	1	`letter+tone`	72.64	79.94	76.11	48.96	42.61	45.57
`S`	LAST	1	`letter+tone`	76.38	75.67	76.02	51.83	45.07	48.22
`M`	BASE	3	`letter`	88.49	66.53	75.96	55.61	62.17	58.71
`M`	BASE	2	`xLH`	81.70	70.64	75.77	53.19	52.49	52.84
`M`	BASE	6	`letter`	74.72	76.62	75.66	48.74	47.31	48.01
`L`	BASE	2	`xLH`	81.24	70.62	75.55	50.59	53.15	51.84
`M`	LAST	4	`letter+tone`	86.32	67.16	75.54	50.11	57.21	53.42
`M`	BASE	2	`xLH`	81.89	69.91	75.42	52.28	52.24	52.26
`M`	BASE	3	`letter+tone`	86.54	66.65	75.30	52.70	57.57	55.03
`L`	BASE	3	`letter+tone`	86.66	66.57	75.30	51.76	58.35	54.86

a non-finite inventory, through the use of a Dirichlet-Process. We used the same hyper-parameters as [7], which were tuned on a larger English corpus and then successfully applied to the segmentation of Mboshi. The best configuration lead to a F-measure on word boundaries (BF) equal to 70.40% which is significantly below the performance of the top-30 configurations reported in Table 4.

Table 5 and 6 brings complementary view on our models' behavior. They compare different models/representations configurations for which F-measures on word boundaries are averaged over spelling model's Markov order and runs. Results in Table 5 are reported for the `S` corpus. They confirm the benefit of keeping tonal information in the representation. As expected, impoverished representations yield worse performance than the `letter+tone` or `letter` representations. However, they still convey enough signal to reliably detect word boundaries. This result is encouraging for future experiments on true speech input, where coarse grain pseudo-phones or pseudo-syllable units could be extracted.

All these results should be taken with some care, given the large standard deviations of results inside each model/ representation combination. The standard deviation is even more important for the results of Table 6 (`L` corpus). One explanation might be the heterogeneous nature of the `L` corpus, but it does not explain all the variability of our runs. The benefit of tone representation is also less clear for the `L` corpus than for the `S` corpus: as more data are considered in training, the usefulness of complex backoff schemes and rich information actually decreases.

In summary, it seems that our models do not take full advantage of the strong signal reflected in the last two lines of Table 3. This might be because these models cannot learn tonal regularities at the grammatical level and are by design limited to learn lexically-based tonal regularities. Yet, tones in Mboshi, as in most Bantu languages, play as much a grammatical role as a lexical one. Related is the current limitation of our models to a unigram word model embedding the more structured spelling model. Bigram dependencies at the word level were consistently shown to improve segmentation [10]. This will be an upcoming extension of the models presented in this work.

Table 5. Comparing different models/representations configurations: F-measures on word boundaries averaged over spelling model's Markov order and runs - `S` corpus

Model	Representation	BF average	std dev	min	max
LAST	`letter+tone`	72.71	3.17	64.97	79.50
BASE	`letter+tone`	69.62	1.82	66.16	73.03
MULTI	`letter+tone`	69.48	3.93	61.46	77.62
BASE	`letter`	69.26	1.98	65.01	73.33
BASE	`xLH`	66.42	3.96	54.43	73.93
BASE	`xV`	63.44	4.48	48.91	71.32
BASE	`CV`	56.14	1.87	52.68	63.47
BASE	`CLH`	50.34	4.19	44.79	60.45

Table 6. Comparing different models/representations configurations: F-measures on word boundaries averaged over spelling model's Markov order and runs - `L` corpus

Model	Representation	BF average	Std dev	Min	Max
BASE	`letter`	69.86	4.22	57.23	77.13
BASE	`xLH`	66.16	6.46	54.45	75.55
LAST	`letter+tone`	65.80	5.94	42.45	73.95
BASE	`letter+tone`	65.40	8.99	43.82	77.95
BASE	`xV`	63.98	7.16	44.12	73.85
MULTI	`letter+tone`	60.94	7.51	46.94	72.86
BASE	`CLH`	55.25	6.25	32.99	63.44
BASE	`CV`	49.90	8.34	23.61	64.85

5 Conclusion

In a preliminary study, we showed that when learning a segmentation classifier on a simplified representation of a Mboshi corpus where all characters were collapsed to two 'vowel' and 'consonant' categories, supplying that classifier with tones provided an increase in performance and even led to a competitive segmentation accuracy despite the considerable simplification of the data. This suggested that segmentation could benefit from sensitivity to tonal cues, and we tried to leverage the latter in an unsupervised setting by introducing hierarchical n-gram spelling models that incorporate tone-conditional distributions in their hierarchy. These models were compared to a standard Pitman-Yor n-gram spelling model for numerous settings. While we were able to observe some benefit in keeping the tonal information in the representation (`letter+tone` vs. `letter`), our proposed spelling model with tones was less successful in capturing tonal signal in unsupervised word segmentation. For this reason, in future work, we hope to exploit tone not purely within the spelling model, but also on the grammatical level beyond simple unigram word sequence models.

Acknowledgments. This work was partly funded by the French ANR and the German DFG under grant ANR-14-CE35-0002. We warmly thank Martine Adda-Decker and Annie Rialland (from LPP-CNRS) for the sketch on the Mboshi language, as well as Gilles Adda (from LIMSI-CNRS), for many meaningful conversations and contributions.

References

1. Adda, G., et al.: Breaking the unwritten language barrier: the Bulb project. In: Proceedings of SLTU (Spoken Language Technologies for Under-Resourced Languages). Yogyakarta, Indonesia (2016)
2. Austin, P.K., Sallabank, J. (eds.): The Cambridge Handbook of Endangered Languages. Cambridge University Press, Cambridge (2011)

3. Beapami, R.P., Chatfield, R., Kouarata, G., Embengue-Waldschmidt, A.: Dictionnaire Mbochi-Français. SIL-Congo Publishers, Congo (Brazzaville) (2000)
4. Bird, S., Hanke, F.R., Adams, O., Lee, H.: Aikuma: a mobile app for collaborative language documentation. ACL **2014**, 1 (2014)
5. Blachon, D., Gauthier, E., Besacier, L., Kouarata, G.N., Adda-Decker, M., Rialland, A.: Parallel speech collection for under-resourced language studies using the LIG-Aikuma mobile device app. In: Proceedings of SLTU (Spoken Language Technologies for Under-Resourced Languages). Yogyakarta, Indonesia, May 2016
6. Bouquiaux, L., Thomas, J.M.C. (eds.): Enquête et description des langues à tradition orale. SELAF, Paris (1976)
7. Godard, P., et al.: Preliminary experiments on unsupervised word discovery in Mboshi. In: Proceedings of the Interspeech (2016)
8. Goldwater, S., Griffiths, T.L., Johnson, M.: Contextual dependencies in unsupervised word segmentation. In: Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, Sydney, Australia, pp. 673–680. Association for Computational Linguistics, July 2006. http://www.aclweb.org/anthology/P06-1085
9. Goldwater, S., Griffiths, T.L., Johnson, M.: Interpolating between types and tokens by estimating power-law generators. In: Advances in Neural Information Processing Systems 18, pp. 459–466. MIT Press, Cambridge (2006)
10. Goldwater, S., Griffiths, T.L., Johnson, M.: A Bayesian framework for word segmentation: exploring the effects of context. Cognition **112**(1), 21–54 (2009)
11. Johnson, M., Demuth, K.: Unsupervised phonemic Chinese word segmentation using adaptor grammars. In: 23rd International Conference on Computational Linguistics (COLING) (2010)
12. Ludusan, B., Synnaeve, G., Dupoux, E.: Prosodic boundary information helps unsupervised word segmentation. In: Annual Conference of the North American Chapter of the ACL, pp. 953–963. Denver, Colorado, USA (2015)
13. Mochihashi, D., Yamada, T., Ueda, N.: Bayesian unsupervised word segmentation with nested Pitman-Yor language modeling. In: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1, pp. 100–108. Association for Computational Linguistics (2009)
14. Rialland, A., Aborobongui, M.E.: How intonations interact with tones in Embosi (Bantu C25), a two-tone language without downdrift. In: Intonation in African Tone Languages, vol. 24. De Gruyter, Berlin, Boston (2016)
15. Stücker, S., et al.: Innovative technologies for under-resourced language documentation: the Bulb project. In: Proceedings of CCURL (Collaboration and Computing for Under-Resourced Languages : toward an Alliance for Digital Language Diversity). Portorož Slovenia (2016)
16. Teh, Y.W., Jordan, M.I., Beal, M.J., Blei, D.M.: Hierarchical Dirichlet processes. J. Am. Stat. Assoc. **101**(476), 1566–1581 (2006)

POS, ANA and LEM: Word Embeddings Built from Annotated Corpora Perform Better (Best Paper Award, Second Place)

Attila Novák(✉) and Borbála Novák

Pázmány Péter Catholic University Faculty of Information Technology and Bionics, MTA-PPKE Hungarian Language Technology Research Group, Práter u. 50/a, Budapest, Hungary
{novak.attila,novak.borbala}@itk.ppke.hu

Abstract. Word embedding models have been popular and quite efficient tools for representing lexical semantics in different languages. Nevertheless, there is no standard for the direct evaluation of such models. Moreover, the applicability of word embedding models is still a research question for less resourced and morphologically complex languages. In this paper, we present and evaluate different corpus preprocessing methods that make the creation of high-quality word embedding models for Hungarian (and other morphologically complex languages) possible. We use a crowd-sourcing-based intrinsic evaluation scenario, and a detailed comparison of our models is presented. The results show that models built from analyzed corpora are of better quality than raw models.

Keywords: Word embeddings · Intrinsic evaluation · Morphologically rich languages

1 Introduction and Related Work

Finding a good representation of words and lexemes is a crucial task in the field of NLP. Neural word embeddings (WE) have proved to be efficient for such tasks [1,7,9].

Various evaluation methods have been proposed for evaluating such models. Some studies use extrinsic evaluation by measuring the effect of using WE's as features in specific tasks. Extrinsic evaluation is important, however, it does not reflect the quality of the embedding model itself, only its effect, which depends on the nature of the downstream task it is utilized in. In most cases, WE models are evaluated in word similarity tasks [1,8] measuring the correlation of the similarity score various embedding models assign to pairs of words to scores assigned by humans (stored in resources called query inventories). However, as Schnabel et al. [15] point out, in this case the problem is that scores from different rankings with different ranges cannot in general be compared, and aggregating such scores may lead to false results. In order to be able to use rank correlation as a metric, a gold similarity of all word pairs in the set including pairs of completely unrelated words would be needed, however, data for only a subset of such pairs is included in these resources. Thus, in order to evaluate and compare WE models in general, they suggest the use of a balanced test set with respect to word frequency, abstractness and part-of-speech.

Most studies focus on the application of embedding models to English or other languages with simple morphology, where the moderate number of different word forms and the relatively fixed word order fit well the theory behind these models. Query inventories are also in general available only for a few languages, mainly English. The dimension along which evaluation is generally performed is the comparison of different implementations of WE models and parameter settings used for training the models on the same corpus. However, in the case of morphologically complex languages, the question of how to handle different word forms of the same lemma has to be considered as well. Our research question is motivated by the fact that, in general, statistical language models built for morphologically complex languages suffer from data sparseness problems: e.g. word-based language models for a morphologically complex language have a notoriously higher perplexity than those for English when created from a corpus of the same size [13].

Ebert and his colleagues experiment with creating lemmatized and stemmed models for several morphologically complex languages (including Hungarian) [3] and conclude that the lemmatized models perform best in word similarity tasks. They introduce a WordNet-based mean reciprocal rank (MRR) metric based on the position of the first item on the nearest neighbor (NN) list retrieved from the embedding model that can be reached from the query word within two steps along some relations present in WordNet. However, the quality and the low and biased lexical coverage of the Hungarian WordNet (HuWN) [6] does not make it a reliable gold standard model of semantic relatedness of words. A high portion of HuWN consists of proper names (83% of the 19400 noun synsets, 38% of all synsets), while the coverage of frequent common words is not very good. Another problem is that HuWN follows the structure of the English (Princeton) WordNet (v2.0) containing many "ghost" synsets that do not have any corresponding word in Hungarian (such nodes make up 6.7% of all synsets). These problems may partly account for the fact that the mean reciprocal rank scores obtained in [3] for Hungarian were so much lower than those obtained for the other languages involved in that experiment. We included the HuWN in our experiments described in Sect. 3 in order to demonstrate that it is not a very good benchmark to compare the performance of WE models against.

The goal of this paper is to investigate the performance of WE models applied to a morphologically complex agglutinative language with free word order, Hungarian, and to perform an exhaustive intrinsic evaluation comparing the effect of different preprocessing scenarios applied to the training set used when building the models. We crucially rely on human evaluation here, to be performed by native speakers in order to get reliable results. This made us limit our investigation to Hungarian.

Table 1. Top NN's of a frequent and a rare word from the SURF and the ANA model. Numbers are corpus frequency.

kenyerek(2270) 'breads' SURF	Vakkalit(5) 'Vakkali.Acc' SURF
kiflik(349) 'bagels'	tevedesnek(5) 'as a mistake'
zsemlék(283) 'buns'	áfa-jának(7) 'of its VAT'
lepények(202) 'pies'	mot-nak(5) 'mot.Dat'
pogácsák(539) 'scones'	Villanysze(5) 'Electrici(an)'
pékáruk(771) 'bakery products'	oktávtól(5) 'from octave'
péksütemények(997) 'pastry.pl'	Isten-imádat(5) 'worship of God'
sonkák(613) 'hams'	Nagycsajszi(5) 'Big Chick'
tészták(2466) 'pasta.pl'	-fontosnak(7) '-as important'
kalácsok(277) 'cakes'	tárgykörbôl(5) 'frôm the subject'
kenyér(147000) 'bread' ANA	Vakkali(23) ANA
hús(136814) 'meat'	Ánanda(321)
kalács(10658) 'milk loaf'	Avalokitésvara(39)
rizs(31678) 'rice'	Dordzse(270)
zsemle(6690) 'roll'	Babaji(82)
pogácsa(11066) 'bisquit'	Bodhidharma(210)
sajt(46660) 'cheese'	Gautama(574)
kifli(9715) 'croissant'	Mahakásjapa(25)
krumpli(37271) 'potato'	Maitreya(426)
búzakenyér(306) 'wheat bread'	Bódhidharma(115)

2 Embedding Models for Hungarian

We built four types of models using the `word2vec` tool using the CBOW model. As a training corpus, we used a 1.2-billion-word raw web-crawled corpus of Hungarian [4]. When building each model, context window was set to 10 words, dimensions to 300, and minimal word occurrence limit to 5. Then we applied different types of preprocessing to the corpus in order to mitigate data sparseness effects due to agglutination. The same strategies can be applied to any other morphologically rich language having a morphological analyzer/tagger/lemmatizer available.

First, we built a model from the tokenized but otherwise raw corpus (SURF). This model is able to represent morphological analogies. E.g. the similarities of the word pairs *jó* – *rossz* 'good – bad' and *jobb* – *rosszabb* 'better – worse' are much higher in this model than if we compare the suffixed form and its lemma, i.e. *jó* – *jobb* 'good – better', and *rossz* – *rosszabb* 'bad – worse'. The first two examples in Table 1 show the nearest neighbors retrieved for some surface word forms. The model represents both semantic and morphosyntactic similarities. The top-n list for *kenyerek* 'bread.plur' has mostly pastries in plural.

However, since due to agglutination, there is a high number of possible surface forms of the same lemma (there are 197 different inflected forms for the lemma *kenyér* 'bread' in the corpus) the model is sensitive to data sparseness effects, since the contexts a word is used in are divided between the different surface forms of the same lemma. The `SURF` model is often not capable to capture the semantics of rare word forms reliably (e.g. the most similar entries for *Vakkalit* 'Vakkali.Acc' are completely unrelated forms in Table 1, col. 2).

Using a morphologically annotated version of the corpus, **we also created a lemmatized model (`LEM`)** in order to bias our model towards representing semantic rather than morphosyntactic and syntactic relatedness and to mitigate data sparseness issues. The annotation was created using the PurePos part-of-speech tagger [12] which also performs lemmatization using morphological analyses generated by the Hungarian Humor morphological analyzer (MA) [10, 11, 14].

Since using lemmata only to build the embedding model may overemphasize semantic relations while suppressing syntactic distributional regularities in the data, **we also created another analyzed (`ANA`) model** following the method described in [16]. A similar method was applied when creating a morpheme-based MT system for Hungarian to solve morphology-related data sparseness problems in [5]. Here we used the same morphologically annotated corpus, but instead of keeping only the lemmata, each word form in the corpus was represented by two tokens: a lemma token followed by a morphosyntactic tag token. The following example shows the representation of the sentence *Szeretlek, kedvesem.* 'I love you, sweetheart.' preprocessed this way:

```
szeret #V.1Sg.>2Sg , #, kedves #N.Poss1Sg
love    V.[I, you] ,     dear    N.[my]
```

Since the tags are kept in the actual context of the word they belong to, the morphosyntactic information carried by the inflections still has a role in determining the embedding vectors. On the other hand, data sparseness is reduced, because the various inflected forms are represented by a single lemma. The second two columns of Table 1 show some examples of top-n lists generated by this model. In contrast to the `SURF` model, the `ANA` model is capable of capturing the semantics of rare lexical items because lemmatization alleviates data sparseness problems and morphosyntactic annotation provides additional grammatical information (compare columns 2 and 4). The most similar entries of *Vakkali* 'Vakkali' in the `ANA` model clearly indicate that the model managed to capture the fact that this is the name of a Buddhist personality.

One of the main drawbacks of raw WE models is that they are not able to handle homonymy, i.e. if a single word form has several distinct meanings, the model assigns a single vector to the word either biased towards one meaning (if that meaning is prevalent in the corpus) or representing a mixture of several meanings. In order to handle this problem at least in those cases, where the different meanings of a word form correspond to different parts of speech, **we also created a modified (`POS`) version** of the `ANA` model keeping the main PoS tag of each word attached to the lemma and only the rest of the morphosyntactic tag was detached. This model assigns a different representation to homonyms having different PoS. The example sentence, *Szeretlek, kedvesem.* 'I love you, sweetheart.' looks like the following in the PoS version of the training corpus:

```
szeret#V #1Sg.>2Sg ,#, kedves#N #Poss1Sg
 love.V     [I, you]  ,   dear.N     [my]
```

3 Experiments

Since no query inventory containing human-assigned word relatedness scores for Hungarian exists that could be used to perform intrinsic evaluation of Hungarian WE models, we decided to follow the method described in [15] and perform comparative intrinsic evaluation of the models setting up a crowdsourcing web page to solicit human evaluation of the different WE models. We asked participants to rank the systems according to relatedness of the words returned by the systems to the query word. In addition to the `ANA`, `POS` and `LEM` models, a modified version of the `SURF` model, an off-the-shelf skip-gram-based model built from a raw corpus (`SGL`) and one based on HuWN were included.

Models built from raw corpora (such as the `SURF` model) contain suffixed word forms, and these appear on the NN lists output by these models. In order to be able to compare the output of these models to those built from preprocessed versions of the corpus, the output of raw models was postprocessed. When nearest neighbors from the original `SURF` model are retrieved for a query word, the result list contains various inflected word of the lemma of the query word or other related words in the top-n list. In order to get the top-n nearest lexemes for the query word instead (and thus make the list comparable to the output of the other models), the resulting list was lemmatized using the MA and repeated occurrences of the same lemma were filtered out, resulting in k nearest lemmata (`SURFL`).

Our models were built from a 1.2-billion-word Hungarian web corpus. There is a freely available model built from a larger, 4.6-billion-token, raw Hungarian webcorpus [17]. While we used the CBOW algorithm to train our models, that model was created using the skip-gram word2vec model, with the default parameter settings (200 dimensions, window size=5). We considered this as a baseline model. We included the lemmatized and filtered output of this model (`SGL`) in our evaluation in order to compare our results to an off-the-shelf model trained using a different algorithm and lower dimensions and narrower window on a larger but different training corpus.

We also included a model derived from HuWN (`WN`). Following the method described in [3], we listed words in the HuWN for each query word that could be reached by an at most 3-steps-long path along any WN relation. The list was sorted by distance from the query word with synonyms of the query word being at distance zero.

3.1 The Set of Queries

A set of query words for English balanced with respect to word frequency, abstractness and part-of-speech were published in [15][1]. This resource included 100 query words. We translated these words (selecting a translation matching the sense supertag and the part of speech given in the original query inventory) and checked the corpus frequency range of the translation. If the translation was not in the same range, then we changed it to another word with the same part-of-speech and semantic category from the proper frequency range. We added 7 words exhibiting homonymy crossing a part of speech boundary, like *vár* ‘castle NN’ vs. ‘wait/expect VB’ or *reggeli* ‘breakfast NN’ vs. ‘morning JJ’.

3.2 The Ranking Task

We asked human annotators to rank the six models (`ANA, SURFL, POS, LEM, SGL, WN`) through a dedicated web interface. In each turn for each annotator, a query word was shown from our list and either 1, 5 or 9 words from each model. The order of the models was randomized in each test case. The result lists were extracted from the k-nearest neighbor lists generated for the query word by each model. The 1, 5 or 9-word-long sequences started either at rank 1 or at rank 30 (in each turn, words from the same range were displayed for each model).[2] There was one exception: in cases when the starting position was 30 for the other models, the `WN` model was left out of the comparison, because the lists generated from HuWN were shorter than 30 items for 97 of the 107 selected words. The coverage of HuWN was rather low anyway: only 70 of the 107 words were covered (65%). If any of the models did not have any result for the given query, an empty list was displayed.

Table 2 shows an example question for the query word *nyúl (főnév)* ‘rabbit (noun)’ presenting a one-word-long list from each model to the annotators[3].

Table 2. An example question asking the annotators to rank the presented words according to their similarity to the query word *nyúl (főnév)* ‘rabbit (noun)’

nyúl (főnév) ‘rabbit (noun)’					
macska ‘cat’	szalad ‘run’	nyúlkál ‘touch’	nyúlféle ‘lagomorph’	malac ‘pig’	dörgölőzik ‘rub’

The annotators had to rank the lists from worst to best according to their intuition of relatedness. For ranking, they could assign a number from 1 to 99

[1] The resource is available online at http://www.cs.cornell.edu/schnabts/eval/.

[2] Although testing at NN rank 30 may seem odd at first, word2vec output even at around rank 2000 often perfectly makes sense. Most entries at around rank 2000 for *macska* ‘cat’ or *nyúl* ‘rabbit’ are animals.

[3] *nyúl* is also a verb in Hungarian ‘reach for’. The verbal sense is 4 times more frequent in the training corpus, dominating the vector representation for most models.

(higher score better) to each system output (0 was given for empty lists by default) and ties were allowed. Annotators were allowed to pass to the next query without ranking when they did not know the word or found ranking the lists too difficult. When evaluating the rankings, absolute values of the scores were ignored. Annotators were also informed in the instructions about the fact that the scores are only used for ranking the lists.

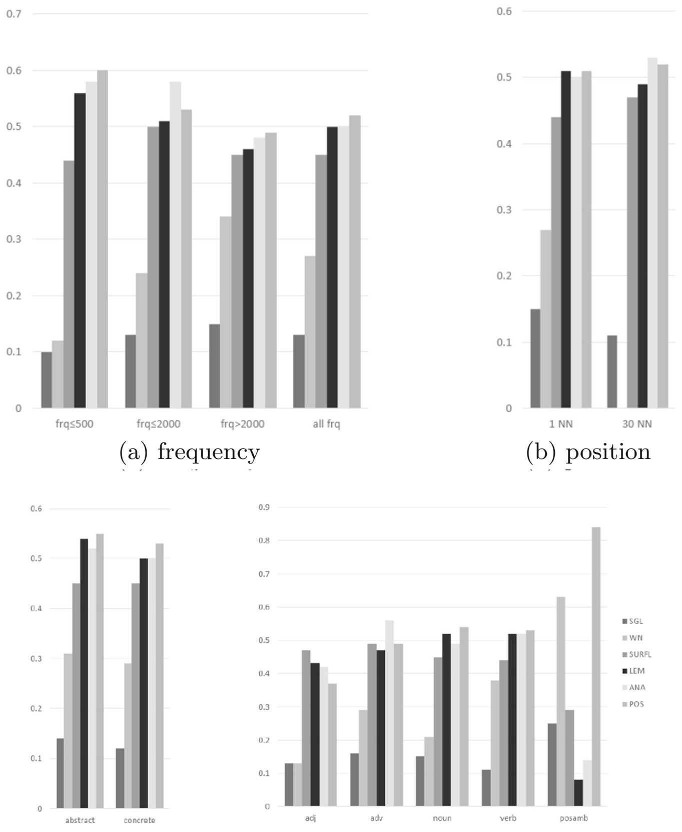
The metric we used for ranking the models was win ratio: the average number of times that the output of a model was judged to be better than another model for the same input. This means the pairwise comparison of the output generated by each model for each query word based on annotator ranking. Each winner scores a point. The final score is wins/comparisons. This method has been used to rank the output of machine translation systems in human evaluation since the beginning of the WMT workshop series, and it has been found to be the most reliable method of comparing the quality of widely differing systems [2]. Rankings were calculated for different values of different parameters, such as the query word frequency range, part of speech, and semantic category (abstractness) of the query word, and the position in the nearest neighbor list (NN 1 or NN 30) where the lists were taken from. The `WN` model was left out of comparison at NN position 30.

Even though the annotators were all native speakers of Hungarian, we introduced a test phase into the system. I.e. each new annotator was given 5 test words randomly placed among the first 10 questions. These test cases contained easy-to-order words and we compared the answers to a gold standard ranking. If an annotator did not manage to reliably distinguish at least distributionally close words and unrelated ones in at least 80% of the test cases, then his or her results were not included in the evaluation. In the end, 15 different annotators provided useful answers each of them submitting 14 to 102 different rankings. We got at least 3 rankings for each query word both at NN position 1 and 30.

4 Results

Results of the evaluation are shown in Fig. 1. One obvious fact that can be seen in the graphs is that the off-the-shelf 200-dimension skip-gram (`SGL`) model [17] performed worst for each condition (p value = 0.05) except for adjectives and very rare words (frequency $\leq$ 500) where `WN` performed just as badly due to lack of coverage, and the PoS ambiguity *(posamb)* condition (see Fig. 1d and its discussion below). NN lists from the `SG` model (i.e. the `SGL` model before lemmatization) contain many "intruders" also for frequent words (e.g. *búzalisztből* 'from wheat flour', *sütéséhez* 'for its baking', *karfiolból* 'from cauliflower' etc. among the top 20 nearest neighbors of *kenyerek* 'loaves of bread'), while the output for rare words often seems to be just all junk. Since the `SG` model was not created by us, we do not know what makes this model perform so poorly (despite the fact that it was created from a significantly larger corpus). It is not probable that the difference is due to the different algorithm (skip-gram vs. CBOW).

Absolute ranking of the models is shown by the *all frq* condition in Fig. 1a. All models we built performed significantly better than HuWN, which confirmed



(a) frequency (b) position (c) abstractness (d) part of speech

Fig. 1. Performance (win ratio) of the models for different conditions.

our doubt about using HuWN as a benchmark for the evaluation of unsupervised embedding resources for Hungarian, its performance being especially dismal for rare words due to the lack of coverage.

As it can be seen in Fig. 1a, building the WE models from an annotated/lemmatized corpus did improve performance for infrequent words (with word form frequency below 500). For frequent words, on the other hand, the performance gain was less pronounced.

When taking the position in the nearest neighbor lists into account (Fig. 1b), the already low quality of the skip-gram model seems to further deteriorate at position 30.

A more interesting result is that the quality of the output of the `ANA` and `POS` models seems to remain higher at NN 30 than that of the `LEM` model. While the lemmatized model seems to handle data sparseness well (see the quality gain at low frequencies), it seems to overrepresent semantic relatedness at the expense of representing grammatical similarity. The NN lists coming from the LEM model are often quite heterogeneous concerning part of speech. The ANA model represents grammatical (syntactic) similarity much better because we kept some representation of the morphosyntactic features in the context when training the embedding vectors for the lemmas. Furthermore, the POS model can even distinguish different senses of homonymous words, as far as ambiguity can be resolved at the level of part of speech tagging. These models seem to be a viable compromise between the word-form-based models built from raw corpora, which, while they represent morphosyntactic knowledge, suffer from data sparseness problems for rare words, and the wildly semantic lemmatized models, which fail to represent much of the (morpho)syntactic information in the data. This is so despite the fact that using the same window size parameter corresponds to only half the context when building the ANA model compared to that used when building the LEM and SURF models[4].

As for abstractness, the relative performance of the models turned out to be almost independent of this dimension (Fig. 1c). The results for different parts of speech can be seen in Fig. 1d. One clear trend is that the models trained on an analyzed/lemmatized corpus clearly performed better for verbs and nouns. This is not surprising, given that these are the most massively inflected categories in Hungarian. The same did not turn out to be true for adjectives: here, the surface model performed better. Note that while adjectives can take the same inflections as nouns in Hungarian in addition to comparatives and superlatives, suffixed forms for adjectives are rare in Hungarian corpora due to the lack of marked agreement within the noun phrase.

The POS model performed at its best when tested on homonymous words one sense of which has a different part of speech than the other (see the *posamb* condition in Fig. 1d). The second best performer was the WN model in this task, as HuWN also contains part-of-speech information. The POS model performed better because it has better coverage. This was the only test where the LEM and ANA models performed worse than both raw-corpus-based models (SGL and SURFL).

5 Conclusion

In this paper, we presented the results of the crowd-sourcing-based intrinsic evaluation of WE models created for Hungarian, a morphologically complex agglutinative language. We built four models with different preprocessing of the same corpus using the CBOW word2vec model. We included two additional

[4] Each word is represented by exactly two tokens in the ANA model, thus the same context window only covers half as many words. The same applies to inflected word forms in the POS model, while noninflected words are represented by only a single token in that model. Note that this effect is mitigated by the fact that word2vec downsamples frequent word forms (among them frequent tags) when creating the model. This corresponds to an effective window size expansion.

models in the evaluation: an off-the-shelf skip-gram model built on a larger raw tokenized corpus and a model based on HuWN.

We found the models utilizing morphological annotations perform better than raw surface-from-based models mitigating data sparseness problems following from agglutination. The novel `ANA` and `POS` models were found to perform even better than the lemma-based model, because, due to the fact that morphosyntactic information is preserved in the context when the models are trained, these models better represent grammatical similarities in addition to semantic ones. Although these models are simple (like Mikolov's CBOW model itself), they perform surprisingly well. The `POS` model is capable of capturing sense distinctions that correspond to PoS distinctions. Our experiments also showed that the quality and especially the coverage of the HuWN resource is not good enough to use it as a benchmark for evaluating Hungarian WE models, as the models themselves were found to perform significantly better under all conditions tested.

Acknowledgments. This research was implemented with support provided by grants FK125217 and PD125216 of the National Research, Development and Innovation Office of Hungary financed under the FK17 and PD17 funding schemes.

References

1. Baroni, M., Dinu, G., Kruszewski, G.: Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 238–247. Association for Computational Linguistics, Baltimore, Maryland (June 2014). https://aclanthology.org/P14-1023.pdf
2. Bojar, O., Ercegovčević, M., Popel, M., Zaidan, O.F.: A grain of salt for the WMT manual evaluation. In: Proceedings of the Sixth Workshop on Statistical Machine Translation, pp. 1–11. WMT 2011, Association for Computational Linguistics, Stroudsburg, PA, USA (2011). https://aclanthology.org/W11-2101/
3. Ebert, S., Müller, T., Schütze, H.: LAMB: a good shepherd of morphologically rich languages. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. Austin, USA (November 2016). https://aclanthology.org/D16-1071/
4. Endrédy, I., Prószéky, G.: A Pázmány Korpusz [The 'Pázmány' Corpus]. Nyelvtudományi Közlemények 112, 191–206 (2016). http://real.mtak.hu/79923/1/NyK20112_u.pdf
5. Laki, L., Novák, A., Siklósi, B.: English to Hungarian morpheme-based statistical machine translation system with reordering rules. In: Proceedings of the Second Workshop on Hybrid Approaches to Translation, pp. 42–50. Association for Computational Linguistics, Sofia, Bulgaria, August 2013. http://www.aclweb.org/anthology/W13-2808
6. Miháltz, M., Hatvani, C., Kuti, J., Szarvas, G., Csirik, J., Prószéky, G., Váradi, T.: Methods and results of the Hungarian WordNet project. In: Proceedings of the Fourth Global WordNet Conference, pp. 311–321 (2008). http://www.inf.u-szeged.hu/~szarvas/homepage/pdf/gwc2008.pdf
7. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. CoRR abs/1301.3781 (2013). https://arxiv.org/abs/1301.3781

8. 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 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5–8, 2013, Lake Tahoe, Nevada, United States, pp. 3111–3119 (2013). https://proceedings.neurips.cc/paper/2013/file/9aa42b31882ec039965f3c4923ce901b-Paper.pdf
9. Mikolov, T., Yih, W., Zweig, G.: Linguistic regularities in continuous space word representations. In: Human Language Technologies: conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, 9–14 June 2013, Westin Peachtree Plaza Hotel, Atlanta, Georgia, USA, pp. 746–751 (2013). https://aclanthology.org/N13-1090
10. Novák, A.: A new form of Humor - mapping constraint-based computational morphologies to a finite-state representation. In: Calzolari, N., et al. (eds.) Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC 2014), pp. 1068–1073. European Language Resources Association (ELRA), Reykjavik, Iceland, May 2014. https://aclanthology.org/L14-1207/
11. Novák, A.: Milyen a jó Humor? [What is good Humor like?]. In: I. Magyar Számítógépes Nyelvészeti Konferencia [First Hungarian conference on computational linguistics]. pp. 138–144. SZTE, Szeged (2003), http://www.morphologic.hu/downloads/publications/na/2003_mszny_Humor_na.pdf
12. Orosz, Gy., Novák, A.: PurePos 2.0: a hybrid tool for morphological disambiguation. In: Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2013), pp. 539–545. INCOMA Ltd., Shoumen, BULGARIA, Hissar, Bulgaria (2013). https://aclanthology.org/R13-1071/
13. Popel, M., Mareček, D.: Perplexity of n-gram and dependency language models. In: Sojka, P., Horák, A., Kopeček, I., Pala, K. (eds.) TSD 2010. LNCS (LNAI), vol. 6231, pp. 173–180. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15760-8_23
14. Prószéky, G., Kis, B.: A unification-based approach to morpho-syntactic parsing of agglutinative and other (highly) inflectional languages. In: Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics. ACL 1999, Stroudsburg, PA, USA, pp. 261–268. Association for Computational Linguistics (1999). https://aclanthology.org/P99-1034/
15. Schnabel, T., Labutov, I., Mimno, D.M., Joachims, T.: Evaluation methods for unsupervised word embeddings. In: Màrquez, L., Callison-Burch, C., Su, J., Pighin, D., Marton, Y. (eds.) EMNLP, pp. 298–307. The Association for Computational Linguistics (2015). https://aclanthology.org/D15-1036/
16. Siklósi, B.: Using embedding models for lexical categorization in morphologically rich languages. In: Gelbukh, A. (ed.) CICLing 2016. LNCS, vol. 9623, pp. 115–126. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-75477-2_7
17. Szántó, Z., Vincze, V., Farkas, R.: Magyar nyelvű szó- és karakterszintű szóbeágyazások [Word-level and character-level embeddings for Hungarian]. In: Tanács, A., Varga, V., Vincze, V. (eds.) XIII. Magyar Számítógépes Nyelvészeti Konferencia, pp. 323–328. Szegedi Tudományegyetem, Informatikai Tanszékcsoport, Szeged (2017). http://acta.bibl.u-szeged.hu/59021/1/msznykonf_013_323-328.pdf

Automatic Normalization of Word Variations in Code-Mixed Social Media Text

Rajat Singh, Nurendra Choudhary(✉), and Manish Shrivastava

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

Abstract. Social media platforms such as Twitter and Facebook are becoming popular in multilingual societies. This trend induces portmanteau of South Asian languages with English. The blend of multiple languages as code-mixed data has recently become popular in research communities for various NLP tasks. Code-mixed data consist of anomalies such as grammatical errors and spelling variations. In this paper, we leverage the contextual property of words where the different spelling variation of words share similar context in a large noisy social media text. We capture different variations of words belonging to same context in an unsupervised manner using distributed representations of words. Our experiments reveal that preprocessing of the code-mixed dataset based on our approach improves the performance in state-of-the-art part-of-speech tagging (POS-tagging) and sentiment analysis tasks.

Keywords: Morphology · Tokenization · Multilingual · Code-mixing

1 Introduction

Code-mixing is the embedding of linguistic units such as phrases, words or morphemes of one language into the utterance of another language, whereas code-switching refers to the co-occurrence of speech extracts belonging to two different grammatical systems. Prevalent use of social media platforms by multilingual speakers leads to an increase in the phenomenon of code-mixing and code-switching [3,5,6,10]. Here we refer both the scenarios as code-mixing. Hindi-English bilingual speakers generate an immense amount of code-mixed social media text (CSMT). [19] noted the complexity in analyzing CSMT stems from non-adherence to a formal grammar, spelling variations, lack of annotated data, inherent conversational nature of the text and code-mixing. Traditional tools presume texts to have a strict adherence to formal structure. Hence, unique natural language processing (NLP) tools for CSMT are required and should be improved upon.

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

A. Gelbukh (Ed.): CICLing 2018, LNCS 13396, pp. 371–381, 2023.
https://doi.org/10.1007/978-3-031-23793-5_30

Internet usage is steadily increasing in multilingual societies such as India, where there are 22 official languages at center and state level, out of which Hindi and English are most prevalent[1]. These multilingual populations actively use code-mixed language on social media to share their opinion. With over 400 million Indian population on Internet, which is predicted to double in next 5 years[2], we notice a huge potential for research in CSMT data.

On analyzing CSMT data, we find following ways in which CSMT data deviates from a formal standard form of the respective language:

- Informal transliteration: These variations are due to lack of a transliteration standard. Multilingual speakers tend to transliterate the lexicons directly from native script to roman, which lacks a formal transliteration method. Hence, it leads to many phonetic variations. For example, can be transliterated to *bahoot*, *bahout* or *bahut*, which may be based on socio-cultural factors like accent, dialect and region.
- Informal speech: These variations are not language specific. Speakers tend to write non standard spellings on social media. Usage of a non-formal speech leads to variations in spellings. For example, cooooooool, gud, mistke or lappy.

Sect. 3 discusses the variations in more detail.

Unless a system captures these variations in code-mixed data, its performance will not be at par with corresponding systems on formal standard texts. In this paper, we present a novel approach of automating the normalization process of code-mixed informal text. We also compare the performances of current state-of-the-art CSMT sentiment analysis [12] and POS-tagging [4] tasks on CSMT data.

Section 2 discusses some relevant recent work and Sect. 3 describes and explains these variations. Section 4 provides information about the datasets we use in this paper. Section 5 gives detailed methodology of our approach. The experiments and evaluations are presented in Sect. 6 and the conclusion is in Sect. 7.

2 Related Work

Analysis of code-mixed languages has recently gained interest owing to the rise of non-native English speakers. [15] normalized the code-mixed text by segregating Hindi and English words. Hindi-English (Hi-En) code-mixing allows ease-of-communication among speakers by providing a much wider variety of phrases and expressions. A common form of code-mixing is romanization [15], which refers to the transliteration from a different writing system to the roman script. But this freedom makes formal rules irrelevant, leading to more complexities in the NLP tasks pertaining to CSMT data, highlighted by [1,2,19].

[1] http://www.thehindu.com/data/sanskrit-and-english-theres-no-competition/article6630269.ece.

[2] https://yourstory.com/2017/06/india-internet-users-report/.

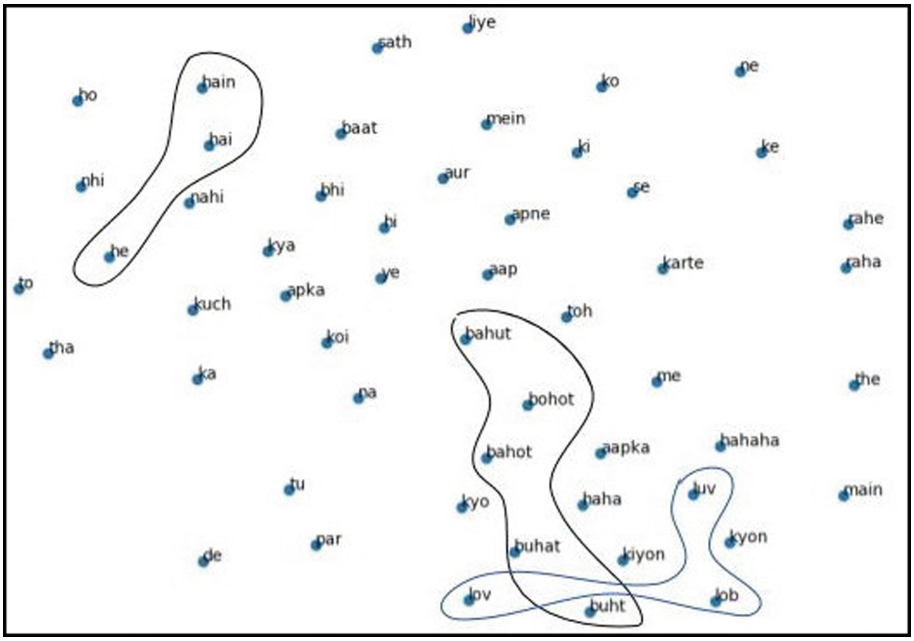


Fig. 1. Most frequent 50 words with some clusters from CSMT data

Initiatives have been taken by shared tasks [14,17] for analysis of CSMT data. [18] used distributional representation to normalize English social media text by substituting spelling mistakes with their corresponding normal form. Deep learning based solutions [16,21] are also demonstrated for various NLP tasks. [15] provides a methodology for normalization of these variably spelled romanized words in a rule-based manner. However, giving an automated unsupervised machine learning model enables the system to be utilized across languages. This is important in case of other Indian Languages, which exhibit similar code-mixing and romanized behavior as Hindi.

3 Types of Variations

Informal variations in lexical forms have not been explored properly. Hence we provide our own nomenclature to better address the normalization process and to provide a discourse for further discussion.

As explained in Sect. 1, we observe informal variations of words in CSMT data. The context we explain the variations here is in Hindi-English per se. However, the approach is not language-specific.

- **Informal transliterations:** Lack of a transliteration standard implies that decisions of marking vowels and other sounds rely entirely on the user. This happens in case of romanizing Hindi which is written phonetically.
  - **Long Vowel transliteration:** Users are found to be indicating vowel length in long vowels in many fashions. For example, the word खाया: is transliterated to *khaaya*, repeating the vowel twice or as *khAya*, using an upper case for the vowel character or just *khaya*, not indicating the length at all. Similar variations observed in साल: *saal*, *sAl*, *sal*,; मेरा: *meraa*, *merA*, *mera*; आजा: *aajaa*, *AjA*, *aja*, *aaja*; or in आपका: *aapka*, *apka*, *Apka*.
  - **Borrowed words' pronunciations**: In case of Indian Languages, especially in Hindi, influence of Persian and English is observed. So when users transliterate foreign words, they rely on the pronunciation when transliterating. For example, some Persian sounds lack an exact counterpart in Hindi, because of which, both इज्जत *ijjat*, and इज़्ज़त *izzat* are observed. Same is observed in आज़ाद *azad* and आजाद *ajad*; ज़िंदाबाद *zindabad* and जिन्दाबाद *jindabad*. This is also observed in English to Hindi transliteration. For example, hospital may be written as (अस्पताल) *aspataal* or (होस्पिटल) *hospital*.
  - **Accent and Dialect Phonetics**: Not all alternate pronunciations arise from different languages, they may be drawn from difference in dialects or accents between users and its transliteration reflects the user's utterance of the word. For example, श्री could be transliterated as both *Shree*, *Sree*, similarly शपत into *shapat* or *sapat*. Certain accents of Hindi speakers lack aspiration. For example, when transliterating खाया to *khaya* or *kaya*.
  - **Double Consonants**: Similar to long vowels, Hindi contains words with double consonants where in speech, stress is given on respective sound. However it is observed that users use variants with or without repeating the respective consonant. For example, इज्जत: *izzat*, *izat* with stress on *z* or in स्वछता: *swachhta*, *swachta* with stress on *ch*.
  - **Non phonetic writing**: Users transliterate differently despite having common pronunciation. The reason is that in English alphabets, multiple characters may have same pronunciation contextually. At the same time, same character may have different pronunciations. For example, k or q may have same pronunciations(king, queue) and hence वक्त could be transliterated as either *waqt* or *wakt*. Similarly in *ee* and *i* pronunciation like in case of *spree* and *ski* is observed. Hence, transliteration of श्री as *Shree* or *Shri* is noticed.
- **Informal speech**: These are variations in spellings which are caused by non formal speech. Following are some variations caused by the informal setting of social media.
  - **Elongation**: Although, speakers have knowledge of word spellings, some spellings are stretched to indicate certain sentiments, like of joy or excitement, word forms like coooooooool, soooooo goood or noooo convey an emphasis which their respective correct spellings can not.

Table 1. Summary of Dataset(Utterances) used for POS-tagging

Language tags	Utterances (Training)	Utterances (Training)
Hindi-English		
English	6178	8553
Hindi	5546	411
Others	4231	2248
Total	15955	11212
Bengali-English		
English	9973	5459
Bengali	8330	4671
Others	6335	3431
Total	24638	13561
Tamil-English		
English	1969	819
Tamil	1716	1155
Others	630	281
Total	4315	2255

- **SMS Language**: An interesting phenomena that is observed in informal social media is of conveying messages in limited number of characters, this style originates from early SMS applications, where messages had a size limit. A noticeable pattern in phrases like *gud nite*, *plz dnt do ths* or *nerndr kb arhn hn*?(in Hindi) is that of dropping most vowels and retaining consonants just enough to guess the word.
- **Abbreviations**: In informal speech, speakers often use abbreviated forms of words, which translates to informal writing. For example, *lappy* for laptop.

Table 2. Summary of Dataset (Sentences) used for POS-tagging

Language	Sentences (Training)	Sentences (Test)
Bengali-English	2837	1459
Hindi-English	729	377
Tamil-English	639	279

Table 3. Statistics of the Hindi-English code-mixed data used to learn distributed representations and Hindi-English code-mixed dataset used for sentiment analysis

Data	Vocab	#Sentences
Twitter Corpus	198025	500000
Hi-En code-mixed data [12]	7549	3879

Table 4. Error analysis of clusters using 500 random sample of words

ϵ	Error %	Average #Words per cluster
0	1.2	3.7
1	8.3	4.5
2	28.1	7.2
3	47.7	11.4

Table 5. Comparison of proposed preprocessing in Sentiment Analysis Task. A, F1 are Accuracy and F1-scores of the models respectively.

Method	Without our approach		Proposed preprocessing	
	A	**F1**	**A**	**F1**
NBSVM (Unigram) [20]	59.15%	0.5335	**60.38%**	**0.5421**
NBSVM (Uni+Bigram) [20]	62.5%	0.5375	**63.43%**	**0.5566**
MNB (Unigram) [20]	66.75%	0.6143	**67.23%**	**0.6325**
MNB (Uni+Bigram) [20]	66.36%	0.6046	**68.85%**	**0.6367**
MNB (Tf-Idf) [20]	63.53%	0.4783	**65.76%**	**0.5025**
SVM (Unigram) [11]	57.6%	0.5232	**58.23%**	**0.5325**
SVM (Uni+Bigram) [11]	52.96%	0.3773	**55.63%**	**0.4238**
Lexicon Lookup [15]	51.15%	0.252	**53.28%**	**0.2745**
Char-LSTM [12]	59.8%	0.511	**60.82%**	**0.5285**
Subword-LSTM [12]	69.7%	0.658	**71.38%**	**0.6621**

4 Dataset

We utilize Twitter as the source of code-mixed text by scrapping 500,000 tweets. We first collect 500 most frequent words in the romanized form of the target language pair (Hindi in our case of Hi-En code-mixed text). Multiple websites showcase most frequent words, including their roman transliterated forms in a particular language. We adopt wiktionary.com[3] for this task. Using these collected most frequent words, we create queries to fetch code-mixed tweets. These queries fetch 500,000 Hindi-English (Hi-En) code-mixed tweets using Twitter APIs[4]. To train a skip-gram model [9] with meaningful vector distribution, we need large data. We remove the punctuations and lowercase the text. These scraped tweets are the CSMT training data for the skip gram model in our preprocessing task. The statistics of this corpus is given in Table 3. This technique easily extends to any code-mixed language pair.

For sentiment analysis task, we consider the dataset (statistics in Table 3) from [12] which is 4981 annotated sentences with 15% negative, 50% neutral

[3] https://en.wiktionary.org/wiki/Wiktionary:Frequency_lists.

[4] https://developer.twitter.com/en/docs/tweets/search/overview.

Table 6. Comparison of proposed preprocessing on POS-tagging

	Without our approach		Proposed Preprocessing	
Language Pair	Baseline (Stanford Model)	CRF Model	Baseline) (Stanford Model)	CRF Model
Bengali-English	60.05%	75.22%	**62.27%**	**76.14%**
Hindi-English	50.87%	73.2%	**53.45%**	**75.24%**
Tamil-English	61.02%	64.83%	**63.38%**	**65.97%**

Table 7. Some example clusters with their respective parent candidate shown in bold.

Standard Form	**Meaning**	**Captured Variations**			
खूबसूरत	beautiful	**khoobsurat**	khubsurat	khubsoorat	khbsurt
क्यूंकि	because	**kyunki**	kiyunki	kiyuki	kyuki
मेहरबानी	clemency	**meherbani**	meharbaani	meharbani	meherbanee
आपका	yours	**aapka**	apkaa	apka	apkA

and 35% positive comments. For POS-tagging of transliterated social media task we adopt dataset from the shared task conducted by Twelfth International Conference on Natural Language Processing (ICON-2015)[5]. Organizers released the code-mixed train and test set for English-Hindi, English-Bengali and English-Tamil language pairs. In Table 1, we provide a summary of the dataset in terms of the utterances. The number of utterances have been recorded for both the training and test data. In Table 2, we present a statistics of the number of sentences for each pair of languages in training as well as test data.

5 Methodology

In this section, we discuss our methodology (shown in Fig. 2) for preprocessing the CSMT data for NLP tasks pertaining to CSMT.

5.1 Skip Gram Distributional Semantic Modeling

We use the collected 500,000 Hi-En CSMT tweets to train Word2Vec embeddings, computed on a 300-dimensional embedding with a skip-gram-10 model [9]. We used the python gensim module [13] to train the representation. We utilized hierarchical sampling for the reduction in vocabulary count during training and used a minimum count of 5 occurrences for each word. This model will be used to extract candidate words and their related variations using clustering. The architecture discussed in this paper can also work with word vectors obtained using other techniques such as latent semantic indexing, convolutional neural networks, recurrent neural networks, etc.

[5] http://ltrc.iiit.ac.in/icon2015/contests.php.

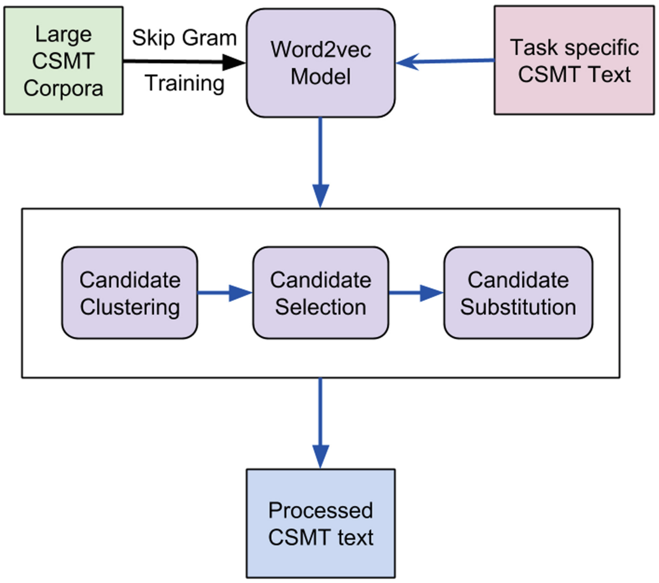


Fig. 2. Methodology

5.2 Candidate Clustering

Skip-gram vectors give the representation of a word in the semantic space based on their context. Due to contextual similarity, variations of the same word will have similar vector representation. Also, it is observed, that the consonants of these variations are similar. Hence, we cluster the words based on a distance metric that captures both these properties. The similarity metric is formally defined below:

$$f(v1, v2) = \begin{cases} sim(vec(v1), vec(v2)) & \text{if EditDistance(v1,v2) } \leq \epsilon \\ 0 & \text{else} \end{cases} \tag{1}$$

where $v1$ and $v2$ are the two variations, sim is a vector similarity function (cosine similarity in our case), $vec(v)$ is a function that returns the skip-gram vector of v, $f(v1, v2)$ represents the overall similarity between $v1$ and $v2$, and the threshold of Levenshtein distance [7] (also known as Edit Distance). ϵ is set according to our requirement of the task. If ϵ is 0 then we will get small clusters with more accuracy but as we increase it further the accuracy will reduce but the coverage will increase with large clusters. We randomly sampled 500 words with their clusters and manually verified the variations of words and calculated the error percentage shown in Table 4. We chose ϵ as 1 for better accuracy and increased coverage. While calculating Levenshtein distance for each pair in clusters we remove the vowels, repetition reduced to single character and digits substituted by their corresponding letters such as *2*→*to* and *4*→*for*. The ϵ varied from 0–2. This metric gives us the closest variations for the given word.

They together form a cluster. Example clusters have been presented in Table 7. Figure 1 showcase most frequent 50 words from the Hindi-English code-mixed large corpora plotted using Matplotlib library[6] and t-SNE[7] from the skip-gram model trained by large code-mixed corpora of Hindi-English language pair. t-SNE [8] helps in the dimensionality reduction to visualize 300 dimensions words.

5.3 Substitution

Finally, in the given CSMT dataset for the task, the parent candidate substitutes each word in their respective cluster. We choose the most frequent word variation as the parent candidate of the cluster. Applying this shows a significant reduction in total unique tokens, and an increase in the frequency of remaining words. Due to this reduction in count of unique tokens by removing the noisy word variations, we demonstrate NLP Tasks performed on the new dataset to be more reliable.

6 Experiments and Evaluation

We tested our approach on the following tasks to evaluate our model

– **Sentiment Analysis on CSMT**: [12] have learned sub-word level representations in LSTM (Subword-LSTM) architecture for sentiment analysis on Hindi-English CSMT. They applied character and sub-word based networks to classify given CSMT text into three sentiments classes - positive, negative and neutral. Performing the same task but along with our preprocessing approach on the data improved the results (illustrated in Table 5).
– **POS-tagging on CSMT**: We tested our preprocessing approach on POS-tagging of CSMT text [4] Hindi-English language pair. [4] used Conditional Random Field, a sequence learning method, to capture patterns of sequences containing code switching to tag each word with accurate part-of-speech information. Introduction of our preprocessing approach in [4] improved the performance (illustrated in Table 6).

7 Conclusions

In this paper, we demonstrate automatic capturing and substitution of different word variations in CSMT data. Our approach exploits the property that words and their variations share similar context in a large noisy text. We also validated our approach by using the methodology on POS-tagging and sentiment analysis of CSMT text which showed improvements over their state-of-the-art counterparts. We demonstrated that informal word variations in CSMT data belong to the same cluster of semantic space and our clustering approach helps in improving the efficiency of finding the candidate words for the substitution

[6] https://matplotlib.org/.
[7] https://lvdmaaten.github.io/tsne/.

phase. This work contributes to the initial step towards building a generic model for automated preprocessing of CSMT data that is valid across large corpora and multiple language pairs. Future extensions of this work could include testing on diverse NLP tasks on CSMT data, as well as across many more language pairs, especially exploring the challenges faced by working on language pairs beyond languages of the Indian subcontinent.

References

1. Barman, U., Das, A., Wagner, J., Foster, J.: Code mixing: a challenge for language identification in the language of social media. In: Proceedings of the First Workshop on Computational Approaches to Code Switching, pp. 13–23 (2014)
2. Chittaranjan, G., Vyas, Y., Bali, K., Choudhury, M.: Word-level language identification using crf: code-switching shared task report of msr india system. In: Proceedings of the First Workshop on Computational Approaches to Code Switching, pp. 73–79 (2014)
3. Duran, L.: Toward a better understanding of code switching and interlanguage in bilinguality: implications for bilingual instruction. J. Educ. Issues Lang. Minority Students **14**(2), 69–88 (1994)
4. Ghosh, S., Ghosh, S., Das, D.: Part-of-speech tagging of code-mixed social media text. In: Proceedings of the Second Workshop on Computational Approaches to Code Switching, pp. 90–97 (2016)
5. Gumperz, J.J.: Discourse strategies, vol. 1. Cambridge University Press (1982)
6. Gysels, M.: French in urban lubumbashi swahili: codeswitching, borrowing, or both? J. Multilingual Multicultural Development **13**(1–2), 41–55 (1992)
7. Levenshtein, V.I.: Binary codes capable of correcting deletions, insertions, and reversals. In: Soviet Physics Doklady, vol. 10, pp. 707–710 (1966)
8. van der Maaten, L.: Learning a parametric embedding by preserving local structure. RBM **500**(500), 26 (2009)
9. 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)
10. Muysken, P.: Bilingual speech: A typology of code-mixing, vol. 11. Cambridge University Press (2000)
11. Pang, B., Lee, L., et al.: Opinion mining and sentiment analysis. Found. Trends Inf. Retrieval **2**(1–2), 1–135 (2008)
12. Prabhu, A., Joshi, A., Shrivastava, M., Varma, V.: Towards sub-word level compositions for sentiment analysis of hindi-english code mixed text. arXiv preprint arXiv:1611.00472 (2016)
13. 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. ELRA, Valletta, Malta, May 2010
14. Sequiera, R., et al.: Overview of fire-2015 shared task on mixed script information retrieval. In: FIRE Workshops, vol. 1587, pp. 19–25 (2015)
15. Sharma, S., Srinivas, P., Balabantaray, R.C.: Text normalization of code mix and sentiment analysis. In: 2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI), pp. 1468–1473. IEEE (2015)

16. Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C.D., Ng, A., Potts, C.: Recursive deep models for semantic compositionality over a sentiment treebank. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pp. 1631–1642 (2013)
17. Solorio, T., et al.: Overview for the first shared task on language identification in code-switched data. In: Proceedings of the First Workshop on Computational Approaches to Code Switching, pp. 62–72 (2014)
18. Sridhar, V.K.R.: Unsupervised text normalization using distributed representations of words and phrases (2015)
19. Vyas, Y., Gella, S., Sharma, J., Bali, K., Choudhury, M.: Pos tagging of English-Hindi code-mixed social media content. In: EMNLP, vol. 14, pp. 974–979 (2014)
20. Wang, S., Manning, C.D.: Baselines and bigrams: simple, good sentiment and topic classification. In: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2, pp. 90–94. Association for Computational Linguistics (2012)
21. Zhang, X., LeCun, Y.: Text understanding from scratch. arXiv preprint arXiv:1502.01710 (2015)

Experiments on Deep Morphological Inflection

Akhilesh Sudhakar(✉), Rajesh Kumar Mundotiya, and Anil Kumar Singh

IIT (BHU), Varanasi, India
akhileshs.s4@gmail.com
http://iitbhu.ac.in

Abstract. Morphological inflection (MI) is the task of generating a target word form based on a source word and a set of target morphological tags. We present different language-agnostic systems for MI and report results on datasets of three different sizes: low, medium and high. All these systems are deep neural network based. We implement and describe a lower baseline, and show that our systems improve on this baseline, as well as meet the state-of-art. One significant contribution through this work is studying the different neural architectures that perform best on different dataset sizes as well as on different languages. Another contribution is exploring the use of phonological features of the language in addition to characters, as well as pre-training of word embeddings. We also implement a hybrid system which combines rules learnt from string alignments along with deep learning. The significance of our work lies in the fact that the systems presented can be used for any language (we present results on 52 languages we experimented with) and in our analysis of how linguistic properties of each language has a strong bearing on the design of the neural architecture used for that language.

Keywords: Morphology · Deep learning · Inflection

1 Introduction

Inflection is the process of transforming a root word (lemma) to a target word form which has a specific set of grammatical features. Common grammatical features include gender, number, tense, aspect etc. The ability to generate an inflected word form from a given lemma (root) is highly beneficial in many NLP tasks. It can be crucial in downstream tasks like topic modeling, machine translation and question answering. Most of these tasks suffer from data sparsity. This is especially true in the case of morphologically rich languages. For instance, in machine translation using parallel corpora, there is always scope for performance improvement provided more words and sentences were available along with their translations in the corpora. Statistically speaking, even then, commonly used words and their inflected forms would repeat and dominate in the corpus. In order to deal with the problem of data sparsity in these tasks, morphological

A. Gelbukh (Ed.): CICLing 2018, LNCS 13396, pp. 382–398, 2023.
https://doi.org/10.1007/978-3-031-23793-5_31

analysis and subsequent morphological inflection could be used. For each word, the root could be extracted using a morphological analyzer. This root will then be translated to the destination language. Morphological inflection will then be applied on the root word (now in the destination language) to obtain the original word but in the destination language. In this way, the actually machine translation system would work on a far smaller subset of words, which consists of just the root words, thereby reducing sparsity. Similarly, morphological inflection can be used in other tasks too [1–3].

An inflected form of a word is generally obtained by the use of affixes (prefixes, infixes, suffixes, simulfixes etc.) which attach morphemes to the beginning, in the middle of or to the end of the root word. For instance, the word *writing* is the present continuous inflected form of the lemma *write*.

We present language-agnostic deep neural systems for morphological inflection. *All the neural networks in these systems condition the target inflected word on the root word and the target features of the inflected word.* The models use sequence-to-sequence modeling of characters in the root word, and generate each character of the target word using the sequence information of characters in the root word as well as the target features. A single neural network uses all the training examples for a language and learns a set of weights for all grammatical features and their tags. This is done by using character-level encodings of root words and binary vector encodings of tags. It is common knowledge that the surface structure of a word and its morphological properties are highly related. These neural systems are built using character-to-character sequence models in order to exploit the surface structure of the root word. We also describe and present a baseline system that does not make use of deep neural networks and show improvements over this baseline.

The current work in an extension of our existing work, presented as part of our participation in the SIGMORPHON shared task at CoNLL 2017 [4]. This work is significantly different from our shared task submission in a number of ways- we build a baseline model, present detailed analysis of hyper-parameter selection, present our findings on using pre-trained vector embeddings, build a hybrid model as well as show the incorporation of phonological features, all of which have not been done in previous work.

There has been considerable debate about the use of deep learning in NLP for low-resource languages. A lot of questions have been raised about the ability of deep learning to work well with low-sized datasets. In order to test the performance of our systems, we present deep learning models trained on datasets of different-sizes (low, medium and high). Our models exceed the baselines on all of these different sizes. However it is also obvious that reducing the size of the training does negatively impact the performance of deep learning systems. In order to overcome this shortcoming, we also present a hybrid model based on extracting rules from string alignments, apart from our neural model. As the size of training data reduces, *we show that the hybrid model shows a growing gain in accuracy over the stand alone neural model.* Pre-training of word vectors also shows considerable performance enhancements.

2 Terminology Used

To maintain a consistent terminology, we use the term ‘root’ for the root word, and ‘target’ for the inflected form of the root word. We use the term ‘feature’ to represent grammatical properties of the inflected word, like, ‘category’, ‘gender’, ‘number’, ‘person’, ‘case’, ‘TAM’, ‘suffix’ etc. We use the term ‘tag’ to represent each of the values taken by a feature for a particular word. For instance, ‘M (male)’, ‘F (female)’ and ‘N (neuter)’ are possible tags for the feature ‘gender’.

3 Background

Prior to neural network based approaches to morphological inflection, most systems used a 3-step approach to solve the problem: 1) String alignment between the lemma and the target (morphologically transformed form), 2) Rule extraction from spans of the aligned strings and 3) Rule application to previously unseen lemmas to transform them. [5–7] used the above approaches, with each of them using different string alignment algorithms and different models to extract rules from these alignment tables. However, in these kinds of systems, the types of rules to be generated must be specified, which should also be engineered to take into account language-specific transformational behavior. [8] proposed a neural network based system which abstracts away the above steps by modeling the problem as one of generating a character sequence, character-by-character. Akin to machine translation systems, this system uses an encoder-decoder LSTM model as proposed by [9]. This model takes into account the fact that the target and the root word are similar, except for the parts that have been changed due to inflection, by feeding the root word directly to the decoder as well. A separate neural net is trained for every language. The results reported by [8] are state-of-art and we match state-of-art on many languages and exceed on some.

Apart from sharing some similarities with [8], our work is unique in many ways. Firstly, the authors in [8] work on a smaller subset of languages and hence report only a limited set of results. Secondly, we detail a hybrid approach to supplement the deep model as well as propose different neural architectures for different sizes of data and for different languages within a particular dataset size too. This has been done taking into account the specific morpho-linguistic properties of these languages. This naturally gives better performance than a single model for all languages. On the other hand, they train a separate neural network for each target feature, whereas we train a single model (specific to a given language and dataset size) on each language. The advantage of doing so is that the model needs lesser data to learn from, because this kind of a model shares weights learned across all the features. This works well because word inflection itself is inherently a mechanism in which the patterns of transformation between a root and a target based on a particular feature’s tag’s contribution are similar to those based on other features’ tags. Training on separate features would not only make the model lose out on making use of this information but would also require more training data to learn from, since sufficient examples need to be present emphasizing the contribution of each of the features. Our work is also novel in presenting the correlation of a language's linguistic properties (based on their occurrence in the phylogenetic tree of languages and based on their morphological complexity) along with the results obtained and the neural architectures designed for each of them. To the best of our knowledge, this is also the only work that additionally makes use of phonological features for morphological inflection.

4 System Description

We present a baseline system as well as two systems for morphological inflection. Each system is tested on all the three dataset sizes. The difference between these two systems is that while the first uses the same neural architecture for all languages (for a particular size of dataset), the second system uses different neural architectures for different languages in order to make best of language-specific morphological information. We present the results of these two systems separately for two reasons. Firstly, the first system can be used to make comparisons across languages because it uses the same configurations for all languages. The first system is more simplistic than the second and provides a higher baseline than the baseline implemented using rule extractions. Secondly, the two systems can be compared with the baseline as well as with each other in order to observe language-dependent variations in performance, which could be insightful for further explorations.

4.1 Broad Basis for Neural Architecture

All the models are deep neural networks that take the root word and the target tag set as inputs. They then generate the predicted target inflected word as output. ***Broadly, these systems treat a word as a sequence of characters and perform sequence-to-sequence treatment of characters in the root word. This sequence modeling allows the model to capture information about other characters in the context of a particular character. This in turn, provides the model with information on how affixes must be introduced into a root word to inflect it***. Moreover, we also observe that the only differences between the root and the target words are those characters that have been added, substituted or deleted due to the inflections. The rest of the characters remain the same in both. Concatenating the character embedding vectors directly with the sequence vector generated by sequence modeling of the characters helps the neural network to take advantage of this fact.

4.2 Common Aspects Across Models

In all the models in both the systems, certain structural and hyper-parametrical features remain the same. The characters in the root word are represented using character indices, while the morphological features of the target word are represented using binary vectors. Each character index of the root word is then embedded as a character embedding of dimension 64, to form the root word embedding, i.e., the root word embedding is a 2-dimensional vector, with each row corresponding to a character position in the word, and the columns of the row corresponding to the embedding of that particular character. If an encoder is used, it is bidirectional and the input word embeddings feed into it. The output of the encoder (if any), concatenated with the root word embedding, feeds into the decoder. All recurrent units have hidden layer dimensions of 256. Over the decoder layer is a softmax layer that is used to predict the character that must occur at each character position of the target word. In order to maintain a constant word length, we use paddings of '0' characters. All models use categorical cross-entropy as the loss function and the Adam optimizer [10].

Dataset. The models were trained on three differently-sized datasets. The low-sized datasets had around 100 training samples, the medium-sized datasets had around 1000 training samples and the high-sized datasets had around 10000 samples for most languages. Datasets were provided for a total of 52 languages. Further details about this dataset, where it was obtained from, and the actual data itself, can be found at the CoNLL-2017 website[1].

Hyper-Parameters. In order to tune the hyper-parameters, we compute a categorical cross entropy error on a held-out validation set. As this tuning would contaminate the validation set and lead to biased estimates, we hold out a separate test set to calculate the model's performance. The reported values are calculated on this test set. Our model has the following hyper-parameters:

1. **Initial learning rate:** We identify this hyper-parameter as the most influential in our model. By this, it is meant that small variations in the initial learning rate led to appreciable changes in accuracies. Given that in our choice of task (inflection), even accuracy changes in the magnitude of 1% can be considered appreciable, we find that the initial learning rate has substantial significance. We use an adaptive learning rate schedule, as suggested by [11]. After experimenting with multiple initial learning rates, we arrive at an initial learning rate of 0.5.
2. **Minibatch size:** While we found that the effect of this hyper-parameter was not so pronounced on accuracy, changes in this hyper-parameter reflected upon the computational time taken during the training phase. Considering that we used a GPU for our experiments, we set this size to 32 to take advantage of matrix computational speedups. We also note that setting this value to anything below 15 does not show any speedups.

[1] https://github.com/sigmorphon/conll2017.

3. **Training epochs:** While our final model uses early stopping to avoid overfitting, we observe that using early stopping masks the effect of the other hyper-parameters. In order to study these more carefully, we initially let our model run all the 10,000 training epochs completely. However, as we have described in the later section on ablation studies, we arrive at a patience value of 10 epochs. Further elaborations have been done in the relevant section.
4. **Momentum:** We find that setting the momentum to 1 gives good results and hence we did not experiment much with this hyper-parameter. This decision was also guided by the intuition that our datasets are not very large and that generally, mostly only unsupervised settings tend to make substantial gains from choosing momentum values with care.
5. **Hidden units:** Dimensions of embedded layer and hidden layers of recurrent units had to be arrived at. We did not observe a progressive trend in increases these dimensions but did observe that the model did marginally better when setting these values to powers of 2. We also note that increasing these above the values chosen (64 for embedding layers and 256 for recurrent layers) did not add to any performance improvements but neither did it degrade performance. However, in the case where we pre-trained word vectors (described in later sections), we observe that increasing the word embedding dimensions to 300 gave better results. We believe that this might be because the word vectors were trained on a Wikipedia dump corpus that naturally has more information than the dataset, which just contains individual words. In order to account for this excess information being passed (as compared to vectors that weren't pre-trained), the model perhaps requires more dimensions.

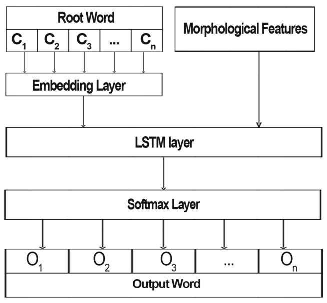


Fig. 1. $C_1, .., C_n$ represent characters of the root word while $O_1, .., O_n$ represent characters of the output word

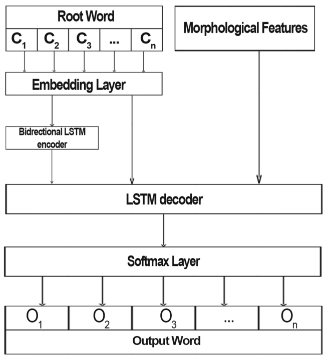


Fig. 2. $C_1, .., C_n$ represent characters of the root word while $O_1, .., O_n$ represent characters of the output word

Low-Sized Dataset. For training the model on the low-sized dataset, we did not use any encoder and we used a simple LSTM with a single layer as the recurrent unit (Fig. 1).

Medium-Sized Dataset. For training the model on the medium-sized dataset, we used a bidirectional LSTM as the encoder and a simple LSTM with a single layer as the decoder (Fig. 2).

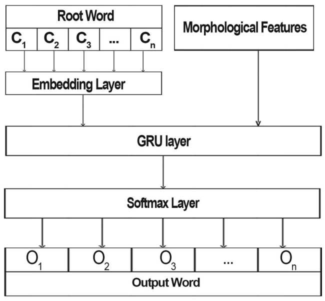


Fig. 3. $C_1, .., C_n$ represent characters of the root word while $O_1, .., O_n$ represent characters of the output word

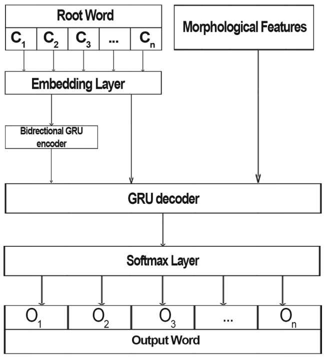


Fig. 4. $C_1, .., C_n$ represent characters of the root word while $O_1, .., O_n$ represent characters of the output word

High-Sized Dataset. For training the model on the high-sized dataset, we used a bidirectional LSTM as the encoder and a simple LSTM with a single layer as the decoder (Fig. 2).

4.3 System 2

Low-Sized Dataset. For training the model on the low-sized dataset, we did not use any encoder and, instead, we used a simple GRU, as reported by [12], with a single layer as the recurrent unit (as shown in Fig. 3).

Medium-Sized Dataset. For medium-sized dataset, we used different model configurations for different languages. Four different kinds of configurations were used:

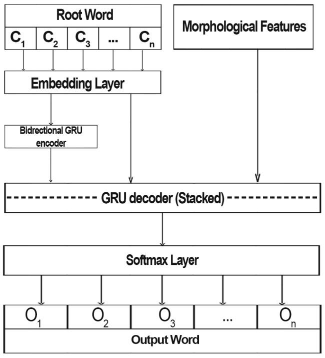


Fig. 5. $C_1, .., C_n$ represent characters of the root word while $O_1, .., O_n$ represent characters of the output word

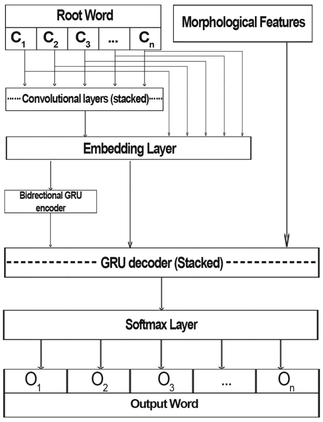


Fig. 6. $C_1, .., C_n$ represent characters of the root word while $O_1, .., O_n$ represent characters of the output word

1. Bidirectional LSTM as the encoder and a simple LSTM with a single layer as the decoder (Fig. 2)
2. Bidirectional GRU as the encoder and a simple GRU with a single layer as the decoder (Fig. 4)
3. No encoder and a simple GRU with a single layer as the recurrent unit (Fig. 3)
4. Bidirectional GRU as the encoder and a deep GRU (two GRUs stacked one above the other) as the decoder (Fig. 5).

High-Sized Dataset. For training the model on the high-sized dataset, we used a bidirectional GRU as the encoder and a deep GRU with 2 layers as the decoder (Fig. 6). Additionally, we use a 2 layer deep Convolutional Neural Network (CNN) to convolve the input word and embeddings. Each convolutional layer uses 64 convolutional filters, each having a width of 5.

4.4 Baseline System

In order to compare our method with a traditional string alignment based method, we use a baseline model as suggested by [13]. However we have made certain modifications to the rule application phase of this baseline model, deviating from the method suggested. This has been done in order to set a more relaxed baseline for low-sized data, given that we are using deep learning techniques, and also to reduce computation time. This model learns a set of rules for each unique combination of target features. For instance, as an example suggested by the authors of the above cited shared task paper, let us assume a training example with root word 'schielen', target word 'geschielt' and has a target feature set

'V.PTCP, PST'. The root and target word are aligned as (based on Levenshtein distance):

___schielen
geschielt__

Next prefix, stem and suffix are extracted from this alignment as follows:

Prefix	Stem	Suffix
___	schiele	n
ge	schielt	__

Based on these pairings, two kinds of rules are extracted: prefix rules and suffix rules. The prefix rule extracted here is to insert the characters 'ge' at the start of the root word. The suffix rules extracted here are to replace the last letter 'e' in the stem of the root word, with the letter 't', and to delete the suffix 'n', of the root word. These 3 rules are now added to other such extracted rules corresponding to the target feature set 'V.PTCP, PST'. Now if a previously unseen test example with root word 'kaufen' and target feature set 'V.PTCP, PST' is encountered, the model searches for a longest match between possible suffixes of 'kaufen' ('n','en','fen', ...) and a root word suffix that exists in the saved rules for the feature set 'V.PTCP, PST'. For instance, if this longest match is 'en' and is part of the rule that replaces 'en' with 't', then the predicted target word for 'kaufen' is 'kauft'.

5 System Enhancements

5.1 Word Vectors and Pre-training

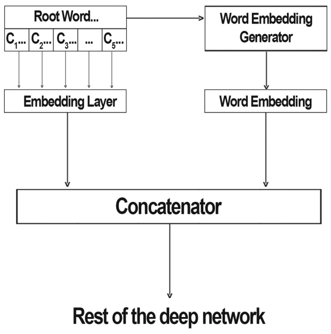


Fig. 7. Incorporation of word vectors

It is not practical to assume that deep learning would give any decent results on datasets of sizes 100 or 1000. Without pre-training, the word vectors are initialized randomly and the onus is completely on the model to train the word vectors. While such training of word vectors entirely in a task-specific manner is known to give good results on supervised learning tasks, the model does a poor job of determining appropriate embeddings with even moderately small datasets. Our analysis shows that the currently obtained results for low and medium sized datasets can be improved by a good margin if pre-trained word embeddings were used. Due to time constraints, we were unable to perform pre-training of word vectors on all the languages that we had datasets for. However, we ran experiments with pre-trained vectors on the datasets for English. Depending on model configurations, we obtained accuracy improvements between 8–10% for low-sized dataset, between 5–7% on medium-sized datasets, and between 0–2% for high sized datasets. We used the model suggested in [14] for the same. One of the main reasons for this performance boost is that the pre-trained word vectors capture syntactic and morphological information from short neighbouring windows. This also means that word vectors can directly capture analogical relations to do with morphology. An example given in [14] is "dance is to dancing as fly is to ___?". Since the task of morphological inflection is concerned with grammatical constructs such as verb tenses, forms of adjectives etc., it is not surprising that enhanced accuracy is observed using pre-training, as this enables the model to draw analogies between training data and previously unseen data. The decrease in accuracy improvement with increase in dataset size is not surprising, as a larger dataset would have already facilitated the generation of better task-specific word embeddings by the model with lesser contribution from pre-trained vectors. Further we also note that languages like Turkish, Czech, Finnish etc. have a lot more to gain from pre-training as compared to English, since they are much more inflectional morphologically complex than English. For instance, in Archi, a single verb lemma can give rise to 1,502,839 distinct forms. The model uses word vectors, incorporated by concatenation with the character vectors, as shown in Fig. 7.

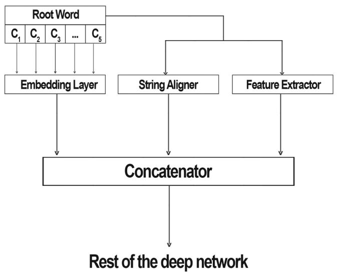


Fig. 8. Hybrid model

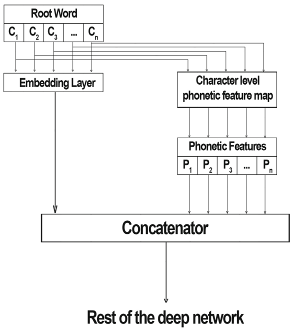


Fig. 9. Incorporation of phonological features

5.2 Hybrid Models

As we have already been highlighted, deep learning as a standalone model is inadequate to handle low-sized data scenarios. This issue becomes important to address as one would ideally like to leverage the advantages that a deep learning model has to offer even in the cases of low-resource languages. Pre-training will only help these languages to some extent, as it too requires a large corpus to learn embeddings from. We explored the use of a hybrid model (Fig. 8) in the case of low-sized and medium-sized datasets on the English dataset. As mentioned earlier, the conceptual framework for morphological inflection consists of *1) String alignment between lemma and target, 2) Rule extraction from spans of these aligned strings and 3) Application of these rule to previously unseen lemmas.* The hybrid system employs traditional methods to address steps 1 and 2, and then feeds the outputs of these traditional methods as inputs to the deep network. Specifically, we borrow from the work of [5] which learns morphological paradigms, by comparing edit operations between the target and the lemma. As an extension to this work, we propose extracting edit rules between the two strings using string alignment algorithms proposed in [5], obtaining a set of features from each lemma-target pair and concatenating the edit rules as well as the features along with the original inputs to the deep net.

1. **String alignment edits:** Rules in this step consist of no-operation, insert, delete and replace. For instance, if the lemma is 'write' and the target is 'had written', one possible set of rules extracted that would have led to each character position of the target string would be as follows: i) insert('h') ii) insert('a') iii) insert('d') iv) insert(' ') v) no-operation vi) no-operation vii) no-operation viii) no-operation ix) insert('t') x) no-operation xi) insert('n') Each of 'no-operation', 'insert', 'delete' and 'replace' are mapped to integers 0–3 respectively (rule types), and a specific rule is encoded as a 3-tuple <rule type, lemma character, target character>. The lemma character and target character are the same for no-operations, the lemma character is an empty character for insert, the target character is an empty character for delete and the target and lemma characters are non-empty for replace rule types.
2. **Word features:** In addition to string alignment rules, we also input word specific features for each character, to the neural network. These are as follows: i) Bi-gram and tri-gram contexts of characters in both lemma and target word ii) Distances of characters in the target word from beginning and end of lemma iii) Distances of characters in lemma from beginning and end of target word iv) Characters' index in both lemma and target word
   (i) and (iv) are especially necessary for low-size dataset configurations, which cannot handle the complexity of an encoder-decoder layer. (ii) and (iv) are required by both medium and low-sized datasets, as these deal with relative with mutual character positions, which the neural network does not handle on its own.

On the low-size dataset, incorporating string alignment rules and word features gave an accuracy improvement of around 1.5% across different configurations. On the medium-sized dataset, the resulting accuracy improvement was around 0.85% across different configurations. The model for the high-sized dataset did not show any improvements. Figure 8 shows the architecture of this hybrid model.

5.3 Phonological Features

It is well known that the phonology of a language interacts closely with its morphology. We used this insight to use phonological or articulatory features as additional inputs to the neural networks. These features are mainly the standard articulatory features such as type (vowel or consonant), place and manner etc. We also use some orthographic features. The details of these features are given in [15]. Each character has a set of 16 phonological features, each having categorical values. The total number of categorical values across all 16 phonological features is 57. The phonological features of each character were encoded in a binary vector of dimension 57, with each position indicating a particular <feature,value> tuple. In order to explore the effect of incorporating character-level phonological features, we ran 2 sets of ablations on the Hindi dataset. In the first experiment, we replaced the characters with phonological features, i.e., each character of a word would be represented by a 57-length binary vector. Doing this gave us an accuracy drop of around 3.4% on the low-sized dataset, 3.2% on the medium-sized dataset and 1.6% on the high-sized dataset. In our second ablation, we concatenated the character-level phonological features along with the character-level embeddings, before passing it to further layers. Doing this improved accuracy on low-sized data by 2.35% and medium-sized data by 0.6% but did not have an effect on high-sized data. Figure 9 shows how the phonological features have been incorporated into the model.

6 Results and Evaluation

Table 1. Accuracies for top-5 languages for low data.

Language	B(A)	B(L)	S-1(A)	S-1(L)	S-2(A)	S-2(L)
Norwegian-Bokmal	69	0.489	52.6	0.71	62.7	0.55
Danish	59.8	0.669	46.1	0.95	49.8	0.87
Urdu	30.3	4.201	31.2	2.48	43.7	1.63
Hindi	31	3.798	33.4	2.34	40.8	2.02
Swedish	54.3	0.884	40.6	1.08	39.4	1.09

6.1 Results on Test Set

The evaluation results were obtained using the evaluation script and the test set provided by the shared task organizers. Baseline accuracies were also obtained

Table 2. Accuracies for top-5 languages for medium data.

Language	B(A)	B(L)	S-1(A)	S-1(L)	S-2(A)	S-2(L)
Quechua	66.2	1.706	93	0.28	93	0.28
Bengali	71	0.44	91	0.19	91	0.19
Portuguese	78	0.103	86	0.21	89.6	0.16
Urdu	81.1	0.287	88	0.47	88	0.47
Georgian	73	0.225	87.7	0.32	87.7	0.32

Table 3. Accuracies for top-5 languages for high data.

Language	B(A)	B(L)	S-1(A)	S-1(L)	S-2(A)	S-2(L)
Basque	6	3.32	100.0	0	100.0	0.0
Welsh	67	0.45	99.4	0.01	100.0	0.0
Hindi	94	0.075	99.3	0.02	100.0	0.0
Portuguese	97.4	0.034	98.5	0.03	100.0	0.0
Persian	77.6	0.567	98.9	0.02	99.9	0.01

from the baseline model provided. The best five baseline accuracies and Levenshtein distances, accuracies and Levenshtein distances for the first submission and accuracies and Levenshtein distances for the second submission can be found in Table 1, Table 2 and Table 3 for each of the three dataset sizes: low, medium and high respectively. In these tables, 'B' stands for Baseline, 'S-1' stands for Submission-1 and 'S-2' stands for Submission-2. 'A' stands for accuracy and 'L' stands for Levenshtein distance.

We have placed the complete set of accuracies and Levenshtein distances for all languages in a Google drive folder[2], sorted by accuracies. The main observation from these tables is that ***languages belonging to the same language family tend to get similar results by our system, which is intuitively valid (although there are many exceptions)***. For example, Romance and Slavic languages tend to occur together in these tables. However, it is not evident from these tables that morphologically more complex languages should be harder to learn, which seems to be counter-intuitive. For example, Turkish is above French. This may be because of hyper-parameters or configurations selected for different languages (which were different, in an attempt to maximize accuracy on the development data).

Figures 10, 11 and 12 show the correlation between accuracy and Levenshtein distance for all three sizes of datasets for submission-1.

The results for the other systems in the shared task that we submitted to, and complete details about the shared task can be found in [13]. The reader may refer to these results to see a comparison of how our system performed.

[2] https://goo.gl/eW46CC.

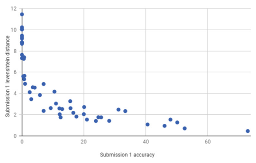


Fig. 10. Accuracy vs. Levenshtein Distance for low data (submission-1)

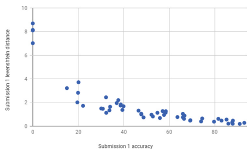


Fig. 11. Accuracy vs. Levenshtein Distance for medium data (submission-1)

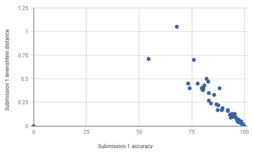


Fig. 12. Accuracy vs. Levenshtein Distance for high data (submission-1)

6.2 Ablation Studies

Early Stop Patience. We observed that for low-sized datasets, both the models (LSTM as well as GRU based) required that at least 10 epochs be run before early stop, every time no progress is detected on the validation set. Setting this patience to less than 5, resulted in near 0 accuracies for most languages and printing of nonsensical target words. For medium-sized datasets, this patience value can be set to around 6–8 while for high-sized datasets, it can be set to around 3–4. However, in order to ensure best results, we set our patience value to 10 across all models, training sizes and languages in the final system.

External Feature Categories. In last year's version of the shared task [16], the morphological features in the dataset were annotated along with the category of each feature. For instance, a sample training feature set from last year is: 'pos=N, def=DEF, case=NOM/ACC/GEN, num=SG'. This year, however, the category of each feature was not provided, i.e., the same example above would appear in this year's format as: 'N,DEF, NOM/ACC/GEN,SG'. Our studies show that while it is conceptually true that the presence of feature categories means exploring a shorter search space, the absence of them does not make a difference to the accuracies obtained for high and medium sized datasets. In the case of low-sized datasets, marginally better accuracies (around 0.5–1%) were obtained when the categories were incorporated into the dataset (this was done manually). However, this might also be the effect of random initialization of parameters.

Choice of Recurrent Unit. Simple Recurrent Neural Networks (RNNs) performed the poorest on all sizes of datasets. For low-sized datasets, in almost all cases, using a GRU gave better results than using an LSTM. On an average, the accuracy increased by 2.33% when shifting from LSTM to GRU as the choice of recurrent unit.

In the case of medium-sized datasets, 8 out of 52 languages performed better with an LSTM than a GRU, while the rest showed better performance with a GRU.

Convolutional Layers. We also ran experiments using convolutional layers, in which the root word was convolved and the convolution was concatenated along with the root word and passed to the encoder layer (if any). The rest of the network structure remained the same. For low-sized and medium-sized datasets, adding convolutional layers resulted in the accuracy dropping to near 0. For high-sized datasets, we were unable to finish running the experiments on all languages due to lack of time. However for the few languages on which we performed convolutional ablation studies, it did seem to improve accuracy by around 1.5% on an average.

Stacking Recurrent Units. Deeper models (more than one layer of LSTM/GRU) resulted in drastic accuracy drops for low-sized datasets. For medium-sized datasets, 30 out of 52 languages showed an accuracy improvement upon stacking two GRU layers, while the accuracy drop in the rest 22 was not drastic but appreciable.

7 Future Work

We aim to perform an exhaustive exploration of language-specific enhancements across all languages in future. Further, we also aim to build separate deep neural networks for different parts of speech (POS) as preliminary experiments show that this gives better performance. In this regard and in the general context of the task of morphological inflection too, we believe that better models can be built by developing a thorough understanding of the morphological nature of each language. While it is true that the techniques described in our work can be used for any language, we hope to customize the configurations by supplementing the word inputs with extra heuristical, rule-based and transduction-based information that are specific to each language. Further insights can be developed by clustering the languages based on accuracy and Levenshtein distance values obtained and correlating these clusters with different classifications. Possible classifications could be done based on the morphological types of languages, namely, analytic, synthetic, agglutinative and polysynthetic. This could not only help in predicting language-specific accuracies for other closely related tasks like morphological disambiguation, but could also help in fine tuning models better for these tasks as well as morphological inflection itself.

8 Conclusions

Different configurations of deep neural networks work well for different languages, based on the morphological properties of the language. It is also true in the context of this task, that phonological features aid the model to perform better. However, standalone deep learning may not be the right approach for low-sized data and hybrid approaches like the one proposed by us give better results. Further, deep learning can be augmented with other transduction, rule-based or knowledge-based methods, to improve on the results we have achieved.

For high-sized data, we achieved accuracies of 100% for some languages. For medium, the highest was 93% and for low, the highest was 69%.

References

1. Minkov, E., Toutanova, K., Suzuki, H.: Generating complex morphology for machine translation. In: ACL, vol. 7, pp. 128–135 (2007)
2. Chahuneau, V., Smith, N.A., Dyer, C.: Knowledge-rich morphological priors for Bayesian language models. In: Association for Computational Linguistics (2013)
3. Oflazer, K., Say, B., et al.: Integrating morphology with multi-word expression processing in Turkish. In: Proceedings of the Workshop on Multiword Expressions: Integrating Processing, pp. 64–71. Association for Computational Linguistics (2004)
4. Sudhakar, A., Singh, A.K.: Experiments on morphological reinflection: Conll-2017 shared task. In: Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection, pp. 71–78 (2017)
5. Durrett, G., DeNero, J.: Supervised learning of complete morphological paradigms. In: Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Atlanta, Georgia, pp. 1185–1195, June 2013
6. Ahlberg, M., Forsberg, M., Hulden, M.: Semi-supervised learning of morphological paradigms and lexicons. In: Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics: Language Technology (Computational Linguistics), Gothenburg, Sweden, pp. 569–578, April 2014
7. Ahlberg, M., Forsberg, M., Hulden, M.: Paradigm classification in supervised learning of morphology. In: Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, pp. 1024–1029, June 2015
8. Faruqui, M., Tsvetkov, Y., Neubig, G., Dyer, C.: Morphological inflection generation using character sequence to sequence learning. In: Proceedings of NAACL (2016)
9. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
10. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. CoRR abs/1412.6980 (2014)
11. Bengio, Y.: Practical recommendations for gradient-based training of deep architectures. In: Montavon, G., Orr, G.B., Müller, K.-R. (eds.) Neural Networks: Tricks of the Trade. LNCS, vol. 7700, pp. 437–478. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-35289-8_26
12. Cho, K., van Merrienboer, B., Gulcehre, C., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. In: Proceedings of EMNLP 2014 (2014)
13. Cotterell, R., et al.: The CoNLL-SIGMORPHON 2017 shared task: Universal morphological reinflection in 52 languages. In: Proceedings of the CoNLL-SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection, Vancouver, Canada. Association for Computational Linguistics, August 2017
14. Pennington, J., Socher, R., Manning, C.D.: Glove: global vectors for word representation. In: EMNLP, vol. 14, pp. 1532–1543 (2014)

15. Singh, A.K.: Digitizing the Legacy of Indian Languages. ICFAI Books, Hyderabad (2009)
16. Cotterell, R., Kirov, C., Sylak-Glassman, J., Yarowsky, D., Eisner, J., Hulden, M.: The sigmorphon 2016 shared task-morphological reinflection. In: Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology, pp. 10–22 (2016)

Identification of Lemmatization Errors Using Neural Models

Attila Novák(✉) and Borbála Novák

Pázmány Péter Catholic University Faculty of Information Technology and Bionics MTA-PPKE Hungarian Language Technology Research Group, Práter u. 50/a., Budapest, Hungary
{novak.attila,novak.borbala}@itk.ppke.hu

Abstract. Most research related to text processing focus on implementing algorithms solving complex tasks, considering simple preprocessing tools as acceptable together with their shortcomings. However, the performance of these low-level tools is sometimes far from being perfect, and errors introduced by them in a text processing chain propagate to higher level modules. In this research, our goal was to create an algorithm that can be used to improve the accuracy of a neglected, but important low-level tool, the lemmatizer. In order to achieve this goal, we experimented with a recurrent neural network classifier and an SVM-based classifier using a word embedding representation of word forms. Our system is able to predict with high accuracy (92.13%) whether a lemma candidate assigned to a word form is correct with the given part-of-speech.

Keywords: Lemmatization · Morphology · Error correction · Machine learning · SVM · RNN

1 Introduction

Most automatic text processing chains consist of a cascade of independent modules, where each module is applied to the output of the preceding one. In the case of such architectures, errors introduced at the beginning of the process, propagate through the entire chain with high-level tools producing erroneous output due to the erroneous input received from earlier modules. Thus it is of crucial importance that tools performing standard preprocessing tasks work as accurately as possible.

In this paper, we propose a method aiming at improving the quality of such a tool: we present the implementation of a method that is able to judge whether a lemma candidate produced automatically by a morphological analyzer/guesser algorithm together with the corresponding part-of-speech tag generated by the tool is correct. Using this method, the number of erroneous lemmata introduced into the annotation during automatic analysis can be decreased significantly.

In our research, we experimented with the lemmatizer algorithm of the PurePos part-of-speech tagger [8]. PurePos provides the possibility of integrating a morphological analyzer (MA), and the morphological analyzer provides lemmata and morphosyntactic tags for words that are included in its lexicon. For those words unknown to the analyzer, however, PurePos applies a simple lemmatization algorithm: a slightly modified version of the suffix guesser algorithm of the TnT tagger [1] is used that returns the operation to be applied to the word ending to obtain the lemma from the original word form in addition to the assigned morphosyntactic tag [8]. The accuracy of this algorithm is relatively low (89.01% measured on standard texts, i.e. it generates one erroneous lemma for every 10 words unknown to the MA [8]).

On the other hand, erroneous lemmata may not only be generated for word forms unknown to the morphological analyzer. In some cases, the analyzer itself overapplies some productive pattern generating an analysis, that may not make sense, with the lemma not being correct either.

The main source of erroneously lemmatized words is noisy or low quality texts. If the input contains a high number of misspelled or non-standard word forms, the morphological analyzer will not be able to cope with them, and, in absence of a morphological analysis, the less accurate suffix based guesser is applied. In the case of misspelled or non-standard words, however, the guesser often has no chance of algorithmically deriving a proper analysis. To make this possible, the text should be corrected prior to analysis.

The aim of our research was to create an algorithm that is able to identify incorrect lemmata, regardless of whether the guesser or the morphological analyzer produced them.

2 Method

We divided lemma candidates produced by PurePos into three classes:

1. Not lemma (`notlem`): the generated lemma candidate is not a lemma, but an inflected word form, e.g. *nyelvészek[N]* 'linguists[Noun]'. The PoS tag is otherwise correct.
2. Bad analysis (`badana`): the generated lemma candidate is not correct considering the assigned part-of-speech, e.g. *vár[Adj]* 'castle/wait[ADJ]'. The lemma candidate itself may or may not be correct. In the latter case, it may contain some orthographic error, or it may be a truncated word form.
3. Correct lemma (`lem`): the generated lemma candidate is correct together with the assigned part-of-speech tag, e.g. *vár[V]* 'wait[Verb]'.

We investigated the performance of two classification algorithms at the task of automatically assigning each lemma candidate to one of these three categories: a character-based recurrent neural network (RNN), and a Support Vector Machine classifier (SVM). The feature set for the SVM was a representation of the lemma candidates that consisted of the embedding vector of the original word form obtained from a raw tokenized corpus, the embedding vector of the lemma and the part-of-speech tag obtained from the morphologically tagged version of the same corpus, and a one-hot representation of the proposed PoS tag itself. In the following subsections, we present each step of the implementation and training of our lemma candidate classification systems as well as evaluation of their performance.

2.1 The Training Set

To create the training set, we used a 1.2-billion-token Hungarian webcorpus [2]. We analyzed the whole corpus using the PurePos tagger augmented with the Hungarian Humor morphological analyzer [5,9]. Being a webcorpus, it contains much noisy, non-standard text (social media contents, comments, blogs, etc.), thus there are many erroneously analyzed and lemmatized words in the output of the tagger/lemmatizer. We collected all the word forms occurring at least 5 times in the corpus together with their analyses. The analyses occurring in the resulting list were either created by the morphological analyzer, or by the guesser algorithm, but no information about the origin of the analysis was present in the output. We applied the morphological analyzer to the word forms appearing in the list and generated two classes of items based on the output of the analyzer. We considered base forms included in the lexicon of the morphological analyzer together with their part of speech as correct lemmata (`lem`). Words that were recognized by the analyzer as correct but inflected word forms rather than base forms were assigned the category (`notlem`).

In addition, we needed to identify analyses that contained incorrect lemmata either due to the form of the lemma or the assigned part-of-speech being incorrect. This type of analysis errors turned out to be especially characteristic of texts containing a high number of word forms with nonstandard spelling, such as unaccented social media texts, erroneously tokenized texts, historical documents etc. In order to collect such badly analyzed words, we first collected lemmata that frequently appeared in their uninflected base forms in the corpus with some part-of-speech, but had an alternative analysis with a different part-of-speech with which they did not appear in the corpus as a base form. For example, the ambiguous lemma *vár*, ‘castle[Noun]’ or ‘wait[Verb]’ also occurred in the annotation generated by PurePos analyzed as an adjective in addition to the correct nominal and verbal analyses. However, only the forms *vár[N]* ‘castle’ and *vár[V]* ‘wait’ appeared as base forms in the output, the adjective analysis was produced by the lemmatizer guesser of PurePos as an erroneous analysis of another word. Starting with just a few dozen such words, we explored the nearest neighbors of these words in a word embedding model built from the analyzed corpus (see the `POS` model in Sect. 2.2), exploiting the earlier observation that words containing similar errors are placed near each other in the word embedding space [6]. Table 1 shows some examples for erroneously lemmatized words and their neighbors in the model. We applied agglomerative hierarchical clustering to the nearest neighbor lists [10] Finally, we created the list used as training data for the category `notlem` by manually checking these sets of neighbors of such incorrect words.

Finally, the training set consisted of 11066 erroneously analysed words, 82582 correct lemmata and 80000 inflected words, from which we separated 1000 randomly selected items from each category as the test set.

Table 1. Examples for incorrectly lemmatized and analysed words, and their nearest neighbours in the (analysed) word embedding model

vár[Adj] ‘wait’	alma[V] ‘apple’	szép[V] ‘beautiful’
tud[Adj] ‘know’	funda[V] [Latin word fragment]	ilo[V] [word fragment]
kér[Adj] ‘ask for’	mangus[V] [Latin word fragment]	nari[V] [word fragment]
jé[Adj] ‘gee’	manda[V] [Latin word fragment]	evian[V] [word fragment]
fogad[Adj] ‘receive’	spiritualité[N] ‘spiritualité’	insa[V] [word fragment]
cso[Adj] [word fragment]	Tibi[V] [fragment of *Tibita*]	manam[V] [word fragment]

2.2 Word Embedding Models

Neural word embedding models place words present in the corpus used for creating them in a few hundred dimensional semantic space, where words with similar meaning and/or syntactic, grammatical or stylistic features are placed near to each other, while less similar words are further apart [3,4]. In this research we used two models described in [7]. One of them was trained on a tokenized but otherwise raw corpus. Words are present in their original, possibly inflected surface forms in this model. In the other case, PoS tagging and morphological analysis was applied to the corpus, and the model was trained on a preprocessed version of the corpus where each original token was represented by one or two tokens. Uninflected words and punctuation are represented by a single token consisting of the lemma concatenated with the main PoS label of the token. In the case of inflected words, this is followed by another token, a complex tag consisting of the rest of the morphosyntactic features of the word (such as case, mood, tense, etc.). The detailed comparison of these models can be found in [7]. Both word embedding models were trained using the CBOW architecture with window size set to 5, and the resulting vectors were 300-dimensional.

2.3 A Character-Based Recurrent Neural Network Classifier

One of the most successful paradigms of recent years is applying neural networks and deep learning for solving different tasks of various fields, such as text processing. Thus, in our present research we also experimented with training a recurrent neural network using the training set for the classification task. The characters of the lemma candidate to be classified are fed to the input layer of the network one by one, followed by the PoS tag as a single token. The characters and the PoS tags are represented as one-hot vectors. The length of these vectors is equal to the sum of the number of different characters and PoS tags found in the training set. Each element of these vectors is 0, except for the element at the index corresponding to the actual character or PoS tag, where the value is set to 1.

The recurrent neural network (RNN) is a two-layer network, with one hidden state. In each step, it uses the vector of the actual input and the result of the preceding step (at the beginning it is set to 0). The output of each step is a 3-element-long vector with the value at each position corresponding to the weight ('probability') assigned by the model implemented in the network to the hypothesis that the input belongs to the given class. This partial result is used as the hidden state for creating the prediction at the next step. The final output is normalized by a LogSoftmax layer resulting in real probability distribution. The architecture of the RNN is shown in Fig. 1. We used the PyTorch framework[1] for implementation with the following parameters: the dimension of the hidden layer was set to 256, the system was trained through 200000 epochs and the learning rate was 0.005.

Given an input in the form of `word[TAG]`, the RNN predicts the probabilities of the input belonging to each class. Even though in the training set an item may occur in more than one class, during evaluation we only considered the top-ranked class selected by the network.

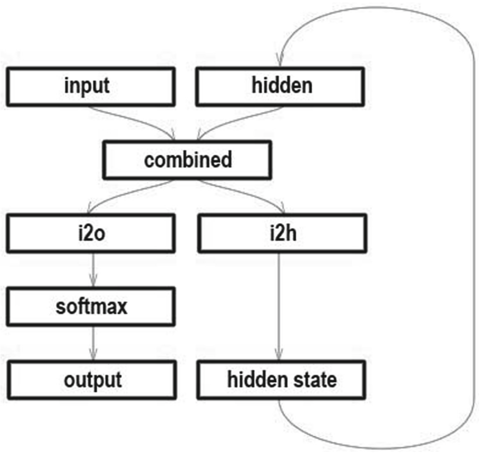


Fig. 1. Schematic structure of a recurrent neural network

2.4 Support Vector Machine Classifier

In another experiment, we used a more traditional classifier using the word embedding vector representation of words as features. A Support Vector Machine (SVM) implements a supervised learning algorithm, where each item is a point in an n-dimensional space and the task is to find those hyperplanes that separate the elements of the different classes best. When using the trained models, each test element is placed in the same space by the model, and this position determines the class assigned to it. The output in this case is a single class, rather than a predicted probability distribution of class membership.

[1] http://pytorch.org/.

The input of the SVM is an n-dimensional vector, constructed as the concatenation of three components:

– the 300-dimensional embedding vector assigned to the form of the lemma candidate (without its PoS tag) by the embedding model created from the raw corpus. If the word is not found in the model, a 300-dimensional vector of zeros is used. This component provides information about the presence of the lemma candidate in the original corpus as an independent word form, and, if it did occur in the corpus, then its vector representation encodes distributional/grammatical characteristics of the word. In general, the lemma of a word is one of its existing forms. Which member of the inflectional paradigm of a word is used as a lemma depends on the part of speech of the word, and is determined by grammatical tradition. In general, a very frequent form is selected. This component is based on two assumptions: first, that the form used as the lemma of an inflected form of a not-very-infrequent lexical item should also appear in the raw corpus, and, second, that there is a systematic relationship between the part of speech of the lexical item and the vector representation of the lemma candidate form both in the case of real lemmata and in the case of other members of the paradigm (i.e. we can assume that the model can learn to distinguish base forms from other inflected forms for each part of speech, and to identify lemma candidates that were assigned the wrong part of speech).
– the 300-dimensional vector representation of the lemma candidate (with its PoS tag) retrieved from word embedding model created from the analyzed corpus. If the word is not found in the model, a 300-dimensional vector of zeros is used.
– the one-hot vector of the PoS tag, as used in the input of the RNN: a vector of length equal to the number of different PoS tags, containing all zeros, except for the index of the actual tag, where it is set to 1. There were 23 different part-of-speech tags in the corpus we used, thus the length of this component was 23.

Thus, the input representation of words for the SVM was a 623 dimensional vector. In order to ensure the possibility of non-linear separation based on the training data, we used the RBF kernel for training the model.

3 Results

For evaluation, 1000 items per class (not present in the training set) were used. Since one word could occur in more than one class, there were some duplicates in the list of randomly selected test words, but for the accuracy of the evaluation, we removed these words. Thus, the number of test words used for evaluating the models was 2974. Quantitative results are shown in Table 2.

We investigated the quality of the classifiers from two aspects. First, we measured the accuracy of the systems for the three-class classification. In this evaluation, only predictions exactly identifying the correct class were considered correct (see *Precision (3-class)*). The three-class classification accuracy of the RNN classifier was 74.37%, and that of the SVM classifier was 87.86%. In both cases, at least about 3/4 of the tested words were classified correctly, but the SVM almost reached 90% precision. Since our initial goal was to create an algorithm that is able to decide whether a lemma with a given part-of-speech is correct, in the other evaluation scenario we did not distinguish the classes `badana` and `notlem`: we did not consider it an error if the model confused these two classes. The `notlem` vs. `badana` distinction is only relevant because texts containing a high number of items from this class can be identified as orthographically substandard. In the latter evaluation scenario, i.e. predicting whether a lemma candidate is correct or not, the precision of the RNN was 80.53%, while that of the SVM was 92.13% (see *Precision (2-class)*). The SVM classifier using word embedding vector representations thus outperformed the character-based RNN in both evaluation scenarios. Even though the word ending plays an important role in determining the morphological class of a word, and the RNN model also had access to orthographic information concerning typical mistyping patterns in the `badana` class, it did not have access to any information about the context each lemma candidate usually appears in, while this information turned out to be both important for the task and effectively encoded in the embedding vectors and efficiently exploited by the SVM.

Table 2. The precision of the two systems in the 3-class and 2-class scenario (`lem` vs. {`notlem|badana`})

	Precision (3-class)	Precision (2-class)
RNN	74,37%	80,53%
SVM	87,86%	92,13%

In a more detailed evaluation, we also quantified the specific types of errors, as shown in Table 3. As it can be seen from the confusion matrices, the RNN assigned more words to the class `badana`, but almost one third of them (31.64%) should have been classified as either an inflected word or a correct lemma. The SVN, however, though having a lower recall, only assigned this class to 5 words erroneously. It can be concluded that the SVM has high precision, i.e. those words that were judged as invalid lemmata could be considered incorrect with high probability. Regarding confusion between the classes `notlem` and `lem`, the number of errors is also smaller for the SVM. 97.8% of those lemma candidates that were judged as not lemma were indeed not lemmata. The SVM model has very good recall for the `notlem` and `lem` classes. Unfortunately, the precision of the correct lemma class is smaller for the SVM, it is 83.3%. Comparing the two systems, the RNN performed better only for the recall of the class `badana`. Items belonging to this class were more often assigned to one of the other classes by the SVM. For the two-class (lemma/not lemma) classification, the SVM model has $R = 96.15\%, P = 83.30\%, F = 89.27\%$.

Table 3. The confusion matrices of the two systems measured on the test set

	RNN	result		
		`badana`	`notlem`	`lem`
gold	`badana`	713	93	156
gold	`notlem`	90	840	70
gold	`lem`	240	113	659

SVM	result		
	`badana`	`notlem`	`lem`
`badana`	646	127	189
`notlem`	0	994	6
`lem`	5	34	973

Looking at the results also revealed that quite often the origin of an incorrect lemmatization was a spelling error. Words that were originally misspelled, and, as a consequence, received an incorrect lemma, were assigned the class `badana` both in the training and the test set, even if the lemma and the PoS tag could have been correct for the correct form of the original word (i.e. the PoS tag is correct and the lemma contains exactly the same spelling error as the original word form). Our algorithm often classified these words as `lem`. This solution could be acceptable in the case of processing noisy texts.

4 Conclusions and Future Work

In this paper, we have presented and compared two algorithms trained to be able to judge whether an automatically generated lemma with a given part-of-speech is correct for a word or not. The implementation using a Support Vector Machine classifier using word embedding features performed better in the task, reaching an accuracy above 92%, while the recurrent neural network could achieve only slightly above 80% accuracy. Both systems were trained and tested for Hungarian, but no language-specific information was used. For training the models, we used a list of words generated automatically and checked manually and we used two types of word embedding representations for each word and their automatically assigned PoS tag. Though our system is not able to suggest a correct lemma for the words it classified as incorrect, if the classifier was integrated into the PoS tagger, the correct lemma could also be retrieved, since the current version of the integrated lemmatizer generates many lemma candidates for words unknown to the morphological analyzer. Instead of blindly selecting the top candidate from this list, the classifier could be used as a filter to remove erroneous candidates.

Acknowledgments. This research has been implemented with support provided by grants FK125217 and PD125216 of the National Research, Development and Innovation Office of Hungary financed under the FK17 and PD17 funding schemes.

References

1. Brants, T.: TnT: a statistical part-of-speech tagger. In: Proceedings of the Sixth Conference on Applied Natural Language Processing. ANLC 2000, pp. 224–231, Stroudsburg, PA, USA. Association for Computational Linguistics (2000). https://doi.org/10.3115/974147.974178, https://aclanthology.org/A00-1031/
2. Endrédy, I., Prószéky, G.: A Pázmány Korpusz [The 'Pázmány' Corpus]. Nyelvtudományi Közlemények **112**, 191–206 (2016). http://real.mtak.hu/79923/1/NyK20112_u.pdf
3. 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 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5–8, 2013, Lake Tahoe, Nevada, United States, pp. 3111–3119 (2013). https://proceedings.neurips.cc/paper/2013/file/9aa42b31882ec039965f3c4923ce901b-Paper.pdf
4. Mikolov, T., Yih, W., Zweig, G.: Linguistic regularities in continuous space word representations. In: Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, 9–14 June 2013, Westin Peachtree Plaza Hotel, Atlanta, Georgia, USA, pp. 746–751 (2013). https://aclanthology.org/N13-1090
5. Novák, A.: Milyen a jó Humor? [What is good Humor like?]. In: I. Magyar Számítógépes Nyelvészeti Konferencia [First Hungarian conference on computational linguistics], pp. 138–144. SZTE, Szeged (2003). http://www.morphologic.hu/downloads/publications/na/2003_mszny_Humor_na.pdf
6. Novák, A.: Improving corpus annotation quality using word embedding models. Polibits **53**, 49–53 (2016). http://www.scielo.org.mx/pdf/poli/n53/1870-9044-poli-53-00049.pdf
7. Novák, A., Novák, B.: Magyar szóbeágyazási modellek kézi kiértékelése [Manual evaluation of Hungarian embedding models]. In: XIV. Magyar Számítógépes Nyelvészeti Konferencia. SZTE, Szeged (2018). http://acta.bibl.u-szeged.hu/59034/1/msznykonf_014_067-077.pdf
8. Orosz, Gy., Novák, A.: PurePos 2.0: a hybrid tool for morphological disambiguation. In: Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2013), Shoumen, Bulgaria, Hissar, Bulgaria, pp. 539–545. Incoma Ltd. (2013). https://aclanthology.org/R13-1071/
9. Prószéky, G., Kis, B.: A unification-based approach to morpho-syntactic parsing of agglutinative and other (highly) inflectional languages. In: Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics. ACL 1999, Stroudsburg, PA, USA, pp. 261–268. Association for Computational Linguistics (1999). https://aclanthology.org/P99-1034/
10. Siklósi, B.: Using embedding models for lexical categorization in morphologically rich languages. In: Gelbukh, A. (ed.) CICLing 2016. LNCS, vol. 9623, pp. 115–126. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-75477-2_7

Full Inflection Learning Using Deep Neural Networks

Octavia-Maria Şulea[1,2], Liviu P. Dinu[1,2], and Bogdan Dumitru[1,2](✉)

[1] University of Bucharest, 14 Academiei, 010014 Bucharest, Romania
[2] Human Language Technologies Research Center, University of Bucharest, Bucharest, Romania
bogdan27182@gmail.com

Abstract. In this paper we present a deep learning architecture capable of predicting the full inflectional paradigms from the uninflected, dictionary form of a word and the full orthographic syllabification. We use Romanian as a case study, since its morphology is rife with stem alternations (non-concatenative inflectional processes) and phonological ambiguity between hiatus and diphthongs. We show that Sequence to Sequence model receiving n-grams as input can solve this problem as good as, if not better than, the state of the art.

Keywords: Computational morphology · Inflection learning · Deep neural networks

1 Introduction and Related Work

Knowing what form to use when learning the representation of a word in an NLP pipeline can be a challenge for languages with rich inflectional morphology [6], where certain inflected forms may be so rare within available corpora that they manifest as hapax legomenons while their corresponding uninflected form might be quite common. This frequency disparity between various forms of the same word leads to vectors of essentially the same underlying concept not ending up near each other in the embedding space, and this in turn leading to wrong conclusions of dissimilarity between the forms. The ability to automatically link all the inflected forms of a word together becomes thus crucial. To this end, research in Computational Morphology has shifted its focus from the traditional construction of rule-based morphological analyzers, which require tedious work to account for alternations in morphologically rich languages, to inflection learning and inflection extraction using machine learning and data-driven techniques.

The automatic induction of full inflections from uninflected, dictionary, forms using machine learning was studied in [8,9,11] and results were presented for Romanian, while the adjacent task of extracting inflection tables in a multilingual setting was introduced in [13] and closely followed by [1] and [14]. In this paper, we look at inflection learning using a deep neural network architecture. While [13] and [9] trained CRF-based models to learn transformation rules

A. Gelbukh (Ed.): CICLing 2018, LNCS 13396, pp. 408–415, 2023.
https://doi.org/10.1007/978-3-031-23793-5_33

between word forms, [1,8,11] used SVMs and [14] used a sequence to sequence character model based on [19]. Although we also used a sequence to sequence model, the input into ours differed from [14] as it only uses the trailing characters representing the difference between inflected and uninflected forms, and the architecture itself differed in several points as presented below.

For inflection learning, we chose to focus on the Romanian datasets ([2,8]) rather than on the multilingual introduced in [13] because the latter contains considerably less base forms for each language (the largest language dataset is for Finish with over 40000 base forms) than the former (over 58000 base forms). We considered that a more varied and *rich* inflection, especially in non-concatenative processes (i.e. apophony), would represent a better test for the effectiveness of a sequence to sequence architecture in inflection learning, since the dataset would give the model less opportunity to overfit particularities of the training data.

Other tasks in Computational Morphology related to inflection learning but leading to benefits in Text to Speech synthesis, *syllabification learning* or *hyphenation prediction*, were investigated as a substitute for complex rule-based systems which were shown to perform poorly in languages containing hiatus-diphthong ambiguities, such as Romanian [12]. From a high-level perspective, automatic hyphenation in ambiguous contexts tests the ability of learning models to distinguish the same sequence in different contexts, while inflection learning for morphologically rich languages tests the ability of these models to identify different sequences in similar contexts. Therefore, inflection learning and automatic syllabification can be construed as complementary tasks. In the present study, we also investigate the use of our proposed system for the task of automatic hyphenation.

2 Romanian Morphology. Alternation and Ambiguity

While Romanian has only one fully irregular verb, the verb *a fi* (to be), it is rife with partial irregularities, both in the stem and in the inflectional suffix patterns. These patterned irregularities, alternations or apophonies (ablaut), occur both in the verbal and nominal domain. Tables 1 and 2 give only a glimpse into the richness of the Romanian inflectional morphology. Extensive tables for the numerous patterns in Romanian inflectional morphology can be found in linguistic studies from [16], who identified 38 types for the verbs alone, [2], who identified 41 verb types, and [7]. As you can see in Table 1, the noun *floare* (flower) has the oa-o stem alternation, whereas the verb *a purta* (to wear) has the u-o-oa alternation (Table 2).

In the case of syllabification, Romanian, along with Portuguese [5], exhibits another interesting feature at the morpho-phonological level: the diphthong-hiatus ambiguity [4]. For instance, the character sequence"ia" (specifically [iV]) can appear in words either as a diphthong, and therefore require no hyphen during character-based syllabification (e.g. piele [pje.le], skin), or as two vowels in hiatus, requiring a hyphen to separate them (e.g. biela [bi.ela], rod). Since Romanian, unlike Portuguese, does seem to rely on consonant clusters around

Table 1. Alternations in the nominal domain

Tag	Regular copac (tree)	Part. Irregular floare (flower)
sg.N-A.indef	copac	fl*oa*re
sg.G-D.indef.	copaci	fl*o*ri
sg.N-A.def.	copacul	fl*oa*rea
sg.G-D.def.	copacului	fl*o*rii
pl.N-A.indef.	copaci	fl*o*ri
pl.G-D.indef.	copaci	fl*o*ri
pl.N-A.def.	copacii	fl*o*rile
pl.G-D.def.	copacilor	fl*o*rilor

Table 2. Alternations in the verbal domain

Tag	Regular a visa (to dream)	Irregular a fi (to be)	Part. Irregular a purta (to wear)
1st sg	vis-ez	sunt	p*o*rt-
2nd sg	vis-ezi	ești	p*o*r*ț*-i
3rd sg	vis-ează	este	p*oa*rt-ă
1st pl	vis-ăm	suntem	p*u*rt-ăm
2nd pl	vis-ați	sunteți	p*u*rt-ați
3rd pl	vis-ează	sunt	p*oa*rt-ă

the vowels in hiatus and the diphthongs to make the distinction [12], we expect this context to help in automatically learning the correct syllabification.

3 Approach

3.1 Dataset and Features

The datasets we used for verbal and nominal inflection learning as well as for hyphenation were based on [3]. In the verbal domain, we extracted over 7000 entries with full inflections and in the nominal domain, around 51000 entries. We split the dataset into test and validation with an 80–20 ratio, for each task. Several prepping steps where applied to the input data, performing proper character alignment, n-gram vocabulary extraction, and padding. One hot encoding was used as a final step in representing the input.

We tried several different feature inputs including character n-grams. The one that performed best for inflection learning was the pair of trailing characters representing the difference between inflected and uninflected forms. For instance, for the verb *a purta* (to wear), the second person singular model was trained on the pair (*urta*, *orţi*) from *purta* (to wear) and porţi (you.SG wear).

For nouns, we initially observed 91 form labels, when taking into consideration gender tags also. This was due to our dataset containing versions of the same nominal base form for two or all three of the genders: masculine, feminine, and neuter. This in turn lead to a lot of variation in the gender tags. We decided to eliminate the gender tags for nouns from the labeling and this way we obtained 8 forms:

1. singular, nominative-accusative, indefinite
2. singular, nominative-accusative, definite
3. singular, genitive-dative, indefinite
4. singular, genitive-dative, definite
5. plural, nominative-accusative, indefinite
6. plural, nominative-accusative, definite
7. plural, genitive-dative, indefinite
8. plural, genitive-dative, definite

For the verbal domain, we focused on the present indicative tense, which is known to be the most irregular [11]. Here, a more intuitive 6 label set emerged:

1. first person, singular
2. second person, singular
3. third person, singular
4. first person, plural
5. second person, plural
6. third person, plural

For the hyphenation learning task, the preprocessing implied extracting character level n-grams (n=6) over the orthographically syllabified words. This generated n+len(word) vectors which were labeled as either 1, if the middle of the n-gram contained a hyphen, and zero otherwise.

3.2 Deep Learning Approach

For inflection learning, we used the *sequence to sequence (seq2seq)* model presented in [19] since this type of model has enjoyed great success in a variety of ML tasks requiring a mapping from one sequence to another. Sequence to sequence learning implies training a model to convert sequences from one domain to sequences from another domain. In our case, we convert the uninflected form of a verb and noun to an inflected form. This is achieved by coupling a so-called encoder with a decoder, feeding the dictionary form to the encoder and expecting the decoder to produce the correct inflected form. For the encoder, a LSTM layer was used to map input data to a fixed sized vector. The resulted vector was then passed to the decoder, implemented using again an LSTM layer to extract the sequence for the vector. A resizing layer was used between encoder and decoder. To improve performance, a proper alignment was used for both input and output data.

For hyphenation, we used an architecture similar to [15]. We implemented these models using Keras[1], wrapping over the Python library for deep learning, TensorFlow[2]. We also used SciPy, NumPy, and sk-learn [18] for the preprocessing steps.

For each inflection learning task and each form, we trained a separate network capable of learning the uninflected-inflected pairing. Therefore, we had 8 trained binary models for the nominal domain, and 6 for the verbal, learning good and bad uninflected-inflected character contexts. The results presented below represent an average of the performance of the 8 and 6 models respectively.

4 DNN Architecture Description

There are several ways to implement a seq2seq model. In our case we have selected a gated RNN layer, more specifically an LSTM, to implement both the encoder and decoder of the model.

At a high level we can view an RNN as a function:

$$y_n = RNN(x_{1:n})$$

where $x_i \in \mathbb{R}^l$ and $y \in \mathbb{R}^m$

The RNN function is defined recursively by a function R that is using a state vector s_{i-1} and x_i as input vector to produce the next state vector s_i. Then, the vector s_i is mapped to y_i using another function O.

We can rewrite the RNN in terms of R and O as follows:

$$\begin{aligned} RNN^*(x_{1:n}, s_0) &= y_{1:n} \\ y_i &= O(s_i) \\ s_i &= R(s_{i-1}, x_i) \end{aligned}$$

where $x_i \in \mathbb{R}^l$, $y \in \mathbb{R}^m$ and $s_i \in \mathbb{R}^m$

An LSTM is a more complex RNN since it has a gated architecture. In fact, it is the first to implement a gating mechanism. When the LSTM was designed, its main functionality was to prevent the vanishing gradient problem from happening.

[1] https://keras.io.
[2] https://www.tensorflow.org/.

Using the above formalism we can now define an LSTM:

$$\begin{aligned}
s_j = R(s_{j-1}, x_j) &= [c_j, h_j] \\
c_j &= f \odot c_{j-1} + i \odot z \\
h_j &= o \odot tanh(c_j) \\
i &= \sigma(x_j W^{xi} + h_{j-1} W^{hi}) \\
f &= \sigma(x_j W^{xf} + h_{j-1} W^{hf}) \\
o &= \sigma(x_j W^{xo} + h_{j-1} W^{ho}) \\
z &= \tanh(x_j W^{xz} + h_{j-1} W^{hz}) \\
y_j &= O(s_j) = h_j
\end{aligned}$$

where $s_j \in \mathbb{R}^{2l}$, $x_i \in \mathbb{R}^m$, $c_j, h_j, i, f, o \in \mathbb{R}^l$, $W^{xo} \in \mathbb{R}^{lm}$ and $W^{ho} \in \mathbb{R}^l$ and i, f, o are gates controlling the input, forgetting, and output. The state vector s_j is split in two vectors c_j and h_j, the first representing the memory component and the second, the hidden state. Gates values are a linear combination between the input x_j and the previous state h_{j-1}. Figure 1 explains this.

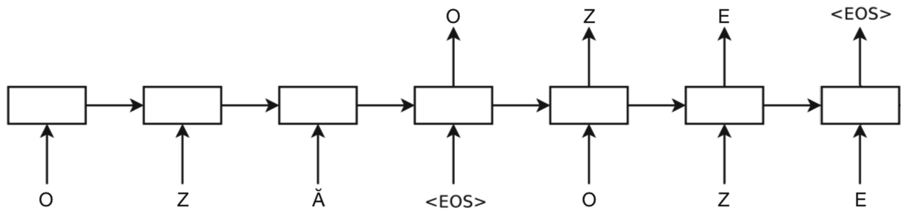


Fig. 1. Out model reads 'OZĂ' sequence (from *poza* - "picture") and produces 'OZE' (from *poze* "pictures") as output sequence. The Seq2seq model stops when the space character is generated.

5 Results

Table 1 shows the results of our model in the verbal and nominal domain and in the syllabification task. For the nominal inflection learning task as well as for hyphenation, our system surpasses the state of the art, while for the verbal inflection learning task our system's performance is on par with the system presented in [9]. For inflection learning, we also compare our results with those in [14] obtained on the Finnish dataset, which, although smaller than our own, it is the closest in number of verb base forms and inflection paradigms. The noun inflection learning results from [14] are given as a rough comparison, since their dataset contained a magnitude less nominal base forms than ours.

Table 3. Accuracy results of our model compared to previous work

Model	Task	Acc
Seq2Seq	Verbal	98.7%
[11]	Verbal	90.64%
[9]	Verbal	98.5%
[14]	FI-V	97.97%
Seq2Seq	Nominal	99%
[8]	Nominal	83.15%
[14]	FI-NA	95.44%
Seq2Seq	Hyphenation	99.5%
[12]	Hyphenation	95%

As we can see, the results show that our system brings significant improvement in the nominal domain and in automatic hyphenation and is on par, with a slight edge, with the state-of-the-art for conjugation learning.

One thing none of the inflection learning systems do is to fully generate the inflected form starting from the uninflected form. For example, our system learns that for the verb *a purta* (to wear) one of its forms, *poartă*, has a u-oa stem alternation and a ta-tă ending transformation, however it was not designed to generate the full inflected form. This is an issue also for the systems published in previous works ([8–11,14]), with only [14] and [8] attempting to move in semi-supervised directions, but not achieving better results than the supervised approaches.

6 Conclusions

In this paper we showed that a sequence to sequence model can be used as a morphological analyzer of uninflected forms and is able to learn full inflections of both verbs and nouns in a morphologically rich language, Romanian. We have also investigated its use in learning end of the line hyphenation, or orthographic syllabification for Romanian. For hyphenation and nominal inflection learning our system outperforms the sate of the art, while for verbal inflection learning, it matches the performance in [9].

Acknowledgment. We would like to thank the reviewers for their useful comments and remarks. The work of Liviu P. Dinu was supported by UEFISCDI, project number 53BG/2016.

References

1. Ahlberg, M., Forsberg, M., Hulden, M.: Paradigm classification in supervised learning of morphology. In: Mihalcea, et al. [17], pp. 1024–1029. http://aclweb.org/anthology/N/N15/N15-1107.pdf

2. Barbu, A.M.: Conjugarea verbelor românîeti. Dicționar: 7500 de verbe româneti grupate pe clase de conjugare. Bucharest: Coresi ,4th ed,, revised. (In Romanian.) (263 pp.) (2007)
3. Barbu, A.M.: Romanian lexical databases: Inflected and syllabic forms dictionaries (2008)
4. Chitoran, I.: The phonology of Romanian. Mouton de Gruyter, A constraint-based approach (2002)
5. Chițoran, I., Hualde, J.I.: From hiatus to diphthong: the evolution of vowel sequences in romance. Phonology **24**, 37–75 (2007)
6. Cotterell, R., Schütze, H.: Morphological word-embeddings. In: Mihalcea, et al. [17], pp. 1287–1292. http://aclweb.org/anthology/N/N15/N15-1140.pdf
7. Şulea, O.M.: Alternations in the Romanian verb paradigm. Analyzing the indicative present. Master's thesis, Faculty of Foreign Languages and Literatures, University of Bucharest (2012). http://ling.auf.net/lingbuzz/001562
8. Şulea, O.M.: Semi-supervised approach to romanian noun declension. In: Knowledge-Based and Intelligent Information Engineering Systems: Proceedings of the 20th International Conference KES-2016, York, UK, 5–7 Sept 2016, pp. 664–671 (2016)
9. Dinu, L.P., Şulea, O.M., Niculae, V.: Sequence tagging for verb conjugation in romanian. In: RANLP, pp. 215–220 (2013)
10. Dinu, L.P., Ionescu, E., Niculae, V., Şulea, O.M.: Can alternations be learned? A machine learning approach to verb alternations. In: Recent Advances in Natural Language Processing 2011, pp. 539–544 (Sept 2011)
11. Dinu, L.P., Niculae, V., Şulea, O.M.: Learning how to conjugate the romanian verb. Rules for regular and partially irregular verbs. In: European Chapter of the Association for Computational Linguistics 2012, pp. 524–528 (Apr 2012)
12. Dinu, L.P., Niculae, V., Şulea, O.M.: Romanian syllabication using machine learning. In: TSD, pp. 450–456 (2013)
13. Durrett, G., DeNero, J.: Supervised learning of complete morphological paradigms. In: Vanderwende et al. [20], pp. 1185–1195. http://aclweb.org/anthology/N/N13/N13-1138.pdf
14. Faruqui, M., Tsvetkov, Y., Neubig, G., Dyer, C.: Morphological inflection generation using character sequence to sequence learning. CoRR abs/ arXiv: 1512.06110 (2015)
15. Fritzke, B., Nasahl, C.: A neural network that learns to do hyphenation. In: International Joint Conference on Neural Networks (1991)
16. Guțu-Romalo, V.: Morfologie Structurală a limbii române. Editura Academiei Republicii Socialiste România (1968) (in Romanian)
17. Mihalcea, R., Chai, J.Y., Sarkar, A. (eds.): NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, 31 May –5 June 2015. The Association for Computational Linguistics (2015)
18. Pedregosa, F., et al.: Scikit-learn: Machine learning in Python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
19. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. CoRR abs/ arXiv: 1409.3215 (2014),
20. Vanderwende, L., III, H.D., Kirchhoff, K. (eds.): Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, 9–14 June 2013, Westin Peachtree Plaza Hotel, Atlanta, Georgia, USA. The Association for Computational Linguistics (2013)

Semi-supervised Textual Entailment on Indonesian Wikipedia Data

Ken Nabila Setya and Rahmad Mahendra(✉)

Faculty of Computer Science, Universitas Indonesia, Depok 16424, West Java, Indonesia
ken.nabila@ui.ac.id, rahmad.mahendra@cs.ui.ac.id

Abstract. Recognizing Textual Entailment (RTE) is a research in Natural Language Processing that aims to identify whether there is an entailment relation between two texts. Textual Entailment has been studied in a variety of languages, but it is rare for the Indonesian language. The purpose of the work presented in this paper is to conduct the RTE experiment on Indonesian language dataset. Since manual data creation is costly and time consuming, we choose semi-supervised machine learning approach. We apply co-training algorithm to enlarge small amounts of annotated data, called seeds. With this method, the human effort only needed to annotate the seeds. The data resource used is all from Wikipedia and the entailment pairs are extracted from its revision history. Evaluation of 1,857 sentence pairs labelled with entailment information using our approach achieve accuracy 76%.

Keywords: Textual entailment · Co-training · Wikipedia revision history · Indonesian language · Semi-supervised · LSTM

1 Introduction

An entailment relation holds between two textual fragments (i.e. text T and hypothesis H) when the meaning of H can be inferred from the meaning of T [9]. Textual entailment recognition is useful to support other Natural Language Processing (NLP) tasks, such as Information Extraction, Question Answering [22], Machine Translation [23], Information Retrieval [8], and Summarization [17].

Not only in English, Recognizing Textual Entailment (RTE) tasks have been also conducted in some other languages, such as Italian [5], Spanish [24], German [30], Arabic [1], and Greek [20]. To our knowledge, no prominent work on RTE task for Indonesian language.

In the data-driven era, textual entailment relation can be identified by machine learning approaches. To obtain good result, the classifier need to learn from more examples in training data. Hickl et al. [13] automatically constructed datasets and successfully improve the performance of textual entailment engines by up to ten percent. The availability of very large scale annotated corpus has stimulated research on natural language inference to achieve accuracy score more than 77% [6].

A. Gelbukh (Ed.): CICLing 2018, LNCS 13396, pp. 416–427, 2023.
https://doi.org/10.1007/978-3-031-23793-5_34

This paper discusses our work on applying semi-supervised method to address the lack of labeled RTE data for Indonesian language. The paper is organized as follows. Section 2 provides information about previous work on creating Textual Entailment data as well as machine learning approach for Textual Entailment recognition. We discuss our proposed methodology in Sect. 3. We report the experiment result in Sect. 4 and conclude in Sect. 5.

2 Related Work

A spectrum of approaches has been proposed for RTE task [2]. Logic-based approach transforms natural language expression into logical meaning representation and then makes reasoning by invoking theorem prover [4,26]. While, machine learning approach learns the set of features from the examples to recognize entailment relationship between pair of text [10,14,15,18,29]. Using maching learning, entailment decision problem can be considered as a classification problem.

Various methods work on different level of language representation, such as surface string, syntactic, or semantic representation. Several measurements can be combined to characterize entailment relation, e.g. lexical overlapping and longest common sub-sequence between surface representation of T and H, tree edit distance of dependency tree representation of input expression, and semantic distance between word meaning across text fragment pair. When applying machine learning approach, such distances or similarity measures can be leveraged as hand-crafted features. In the recent years, deep neural networks successfully deliver remarkable result to the NLP tasks, including RTE. LSTM model has outperformed similarity-based or traditional machine learning-based classifiers to tackle the RTE task [6,25,27].

Experimenting with textual entailment recognizer requires the datasets containing both positive and negative input pairs. Commonly used data for the evaluation of textual entailment and natural language inference task for English language are dataset from The PASCAL Recognising Textual Entailment Challenge [9,11,12]. Larger benchmark is the data for SemEval 2014 shared task, called Sentences Involving Compositional Knowledge (SICK) that consists of 10,000 sentence pairs; each annotated for relatedness in meaning [19]. Bowman [6] issues the Stanford Natural Language Inference (SNLI) corpus, a collection of 570K sentence pairs labeled for entailment, contradiction, and semantic independence.

While the annotation effort of RTE data is done by the experts, the annotation of SICK and SNLI dataset relies much on manual labor via crowdsourcing system. Apart from human intervention in collecting and labeling data, several works attempt to acquire data (semi-)automatically. Burger generates an entailment recognition corpus from the news headline and the first paragraph of a news article. However, this approach is limited to grab the positive entailment pairs only [7]. Hickl improved upon this work by including negative examples using heuristic rules (e.g., sentences connected by `although`, `otherwise`, and `but`) [13].

Zanzotto conducted the experiments using the co-training method on Wikipedia data to expand the Textual Entailment corpus [28]. Wikipedia revision history can provide two views that are required for the application of Co-training methods. The two views used in their research are the pairs of texts before and after revision and the author comments when revising the article page. The first view of data is represented in syntactic features to be classified using the SVM-light classifier. Whereas, the second view is represented by the bag-of-word.

Our proposed method to approach RTE problem for Indonesian language is semi-supervised. To expand the data from small amount of annotated data, we adopt co-training algorithm from Zanzotto work with slight modification. Instead of using syntactic parse tree (which has not been extensively and carefully studied in Indonesian language) to engineer the features for the classifier, we play with embedding vector representation of words in text pairs that are trained in LSTM model.

3 Methodology

We utilize Indonesian Wikipedia as the source of Textual Entailment dataset. Wikipedia all articles and revision history data in XML format are downloaded from Wikimedia dump site[1] The text contents are obtained after cleaning Wiki markup language format from Wikipedia all articles using Wikipedia Extractor tool[2] The texts are segmented into list of sentences by rule-based sentence splitter. A collection of sentences is pre-trained to construct word embedding model using the word2vec [21]. On the other hand, Wikipedia's revision history is taken to produce candidate of entailment pairs. The whole picture of the methodology of this study can be seen in the Fig. 1.

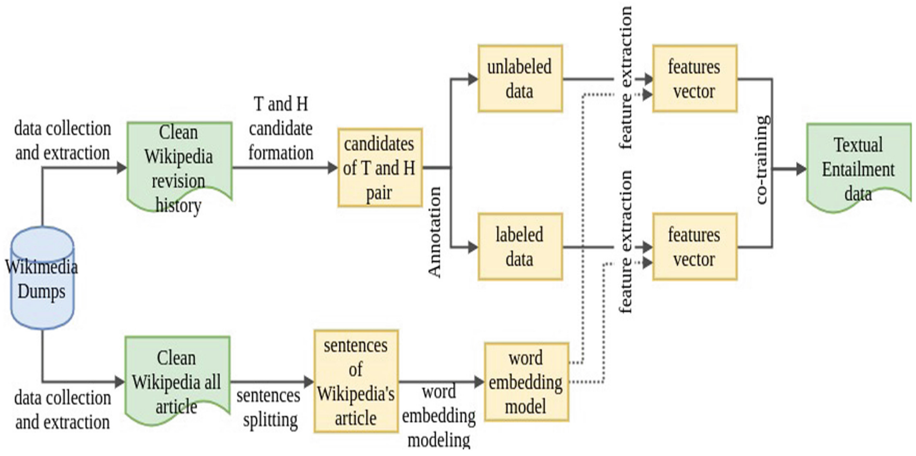


Fig. 1. Proposed Methodology

[1] https://dumps.wikimedia.org/.
[2] https://github.com/attardi/wikiextractor.

3.1 T and H Candidate Generation

Wikipedia's revision history stores all changes made to any article page. Each revision has its own identifier and the parent identifier (the identifier of previous version of the article). We collect pairs of texts before and after revision for each revision. For each pair, we also extract the comment post that the author wrote when revising Wikipedia article page.

Preprocessing Vandalism Edits. Since Wikipedia is written collaboratively, the content quality may be vulnerable. Whereas some revisions of Wikipedia pages are for constructive purpose (i.e. either inserting new information to the text or deleting old information), several edits are vandalisms that deliberately compromise Wikipedia's integrity. At least, vandalism edits are characterized in two ways: (1) the occurence of vulgar and abusive languages, interjections to scold or to make a joke, and/or emoticons. For example: sentence *Chairil Anwar adalah seorang penulis puisi hahahaha* (en: Chairil Anwar is a poet hahahaha) and (2) falsifying the information by changing the meaning. For example: sentence *Andi Hermanto ganteng adalah salah satu mantan presiden Indonesia* (en: The handsome Andi Hermanto was an ex-president of Indonesia). In fact, a man named Andi Hermanto has never be a president.

In the context of our task goal, the first type of vandalism edit needs to be pre-processed. A dictionary of abusive words is compiled. When any word in this dictionary is found in the Wikipedia text revision, it is replaced by the empty string. On the other hand, the text fragments belonging to the second type of vandalism edit is retained as it is. The latter type can be potential candidate of negative entailment pair.

Sentence Alignment and Pairing. For a particular revision pair, suppose that the text before revision consists of m sentences $(b_1, b_2, ..., b_m)$ and the text after revision consists of n sentences $(a_1, a_2, ..., a_n)$. We can align two sentences b_i $(1 \leq i \leq m)$ and a_j $(1 \leq j \leq n)$ if both are identical or differentiable by spelling variation. We apply edit distance algorithm to detect typo errors.

The sentence b_i that does not have any alignment is the sentence that previously exists and then deleted after the revision process. While, the sentence a_j that is not aligned with any sentence is introduced by author when revising the article page. Those sentences without alignment are considered as component of entailment pair following three assumptions below.

1. The sentence deleted from the text before revision are paired with newly-added sentences in the text after revision. We infer that new sentence substitutes information of deleted sentence.
2. If the number of deleted sentences (i.e. m) is more than the number of added sentences (i.e. n, $m > n$), then only first n deleted sentences are aligned with added sentences after revision (b_1 is paired with a_1, b_2 with a_2, ..., b_n with a_n). We think that the rest deleted sentences $(b_{n+1}, ..., b_m)$ can be omitted by the author because it is unimportant or other undesirable cases.

3. If the number of added sentences (i.e. n) is more than the number of deleted sentences (i.e. m, $m < n$), then first m added sentences are aligned with deleted sentences before revision (b_1 is paired with a_1, b_2 with a_2, ..., b_m with a_m). In addition, the rest of added sentences are paired with previous sentence (i.e. a_{m+1} is paired with a_m, a_{m+2} with a_{m+1}, ..., a_n with a_{n-1})

Figure 2 illustrates the sentence alignment and pairing process. To determine which sentence is T and which one is H in the entailment relation T $\rightarrow$ H, we compare the length of sentence before and after revision. The longest one is assigned as T.

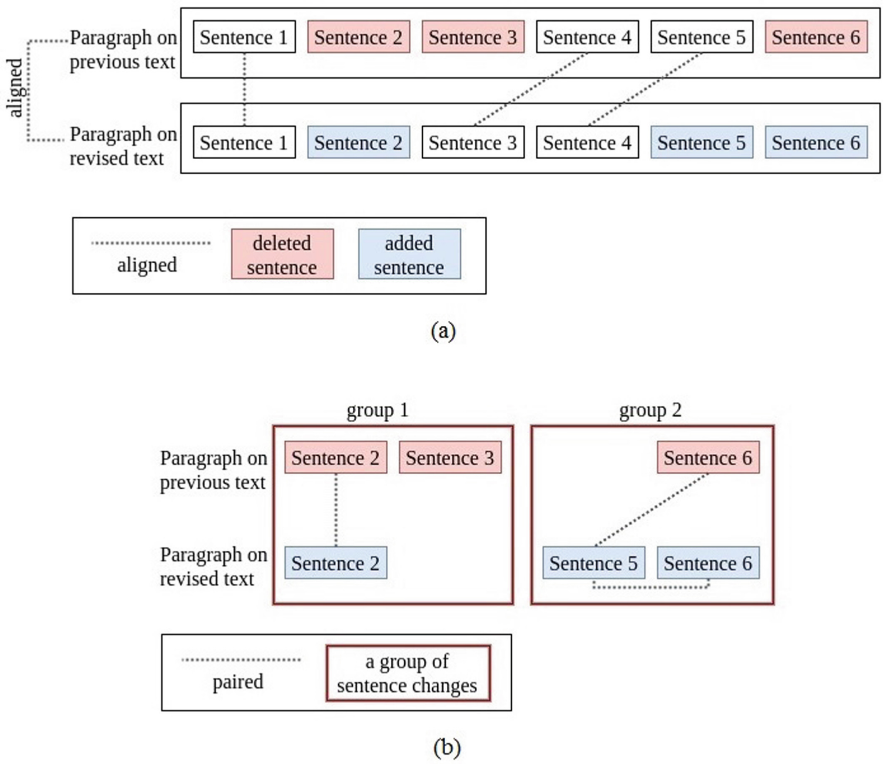


Fig. 2. Sentences alignment and pairing to obtain T and H pair candidate

3.2 Seed Annotation

Manual annotation is performed on a small portion of the T and H pair candidates. 400 pairs are randomly chosen and each of them is judged by three different annotators. The Kappa agreement [16] of annotation result is 0.827.

In the final data that has been validated, we have 223 pairs annotated with E (entailment) label and the rest, 177 pairs, are with NE label (not entailment) label. Annotated data is used as the initial seed for the semi-supervised process.

3.3 Entailment Classification

The co-training [3] method needs two views that are conditionally independent. We assign the candidate pairs of T and H as the first view and author's comment post as the second view.

We implement LSTM model following Bowman architecture [6] as the classifier on the first view of our data. We apply word embedding as the features. Our proposed model architecture is described in Fig. 3.

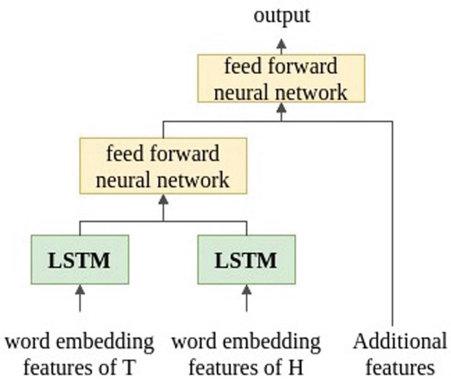

output
feed forward neural network
feed forward neural network
LSTM
LSTM
word embedding features of T
word embedding features of H
Additional features

Fig. 3. Model architecture of first view

We augment some additional features beside word embedding of T and H sentences to strengthen lexical relationships between T and H. To extract additional features, first we apply words alignment process between T and H pairs. We align identical word occurring in both T and H parts. The rest of words in each part are clustered based on neighboring aligned word. We then align the corresponding cluster between T and H pairs. There is possibility that a particular cluster in one part (T or H) is aligned to empty cluster in another part. The additional features that we use in our model are as follows.

1. The similarity of embedding vector of corresponding non empty clusters between T and H pairs.
2. The number of cluster(s) in T paired with an empty cluster in H, and vice versa.
3. The number of identical words aligned between T and H pairs.
4. The longest sub-sentence of T and H.
5. The boolean value indicating that T and H begin with the same word.

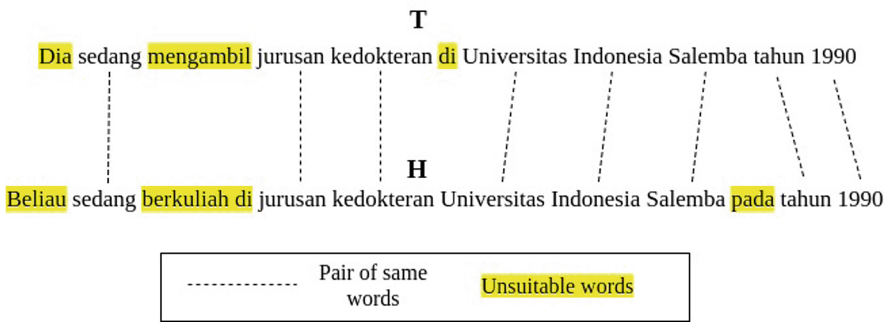

T
Dia sedang mengambil jurusan kedokteran di Universitas Indonesia Salemba tahun 1990
H
Beliau sedang berkuliah di jurusan kedokteran Universitas Indonesia Salemba pada tahun 1990
Pair of same words
Unsuitable words

Fig. 4. Example of words alignment

Figure 4 illustrate an example of engineering process of additional features that is used by the classifier on first view.

The aligned words in aforementioned example are *sedang, jurusan, kedokteran, Universitas, Indonesia, Salemba, tahun,* and *1990*. There are 4 unsuitable word clusters: $\{\{dia\}, \{beliau\}\}$, $\{\{mengambil\}, \{berkuliah,\ di\}\}$, $\{\{di\}, \{\}\}$, and $\{\{\}, \{pada\}\}$.

1. The similarity of non-empty clusters of unsuitable word between T and H. The calculation of this feature for the example is

$$\frac{(sim(\{dia\},\{beliau\})+sim(\{mengambil\},\{berkuliah,\ di\})}{2}$$

2. A value in the range 0 to 1 that describes the number of non-empty group(s) in T paired with an empty group in H. To produce values in the range of 0 to 1, the number can be incorporated into the sigmoid function. Since there is only one pair, that is $\{\{\}, \{pada\}\}$, the value of this feature for given example is $sigmoid(1)$
3. A value in the range 0 to 1 that describes the number of non-empty group(s) in H paired with an empty group in T. The value of this feature for given example is $sigmoid(1)$ as only one pair exists, i.e. $\{\{di\}, \{\}\}$
4. A value in the range 0 to 1 that represents number of identical words aligned between T and H pairs. To obtain a value between 0 and 1, the number of words occurring in both T and H should be divided by the total number of words in the text T or H. There are eight pairs of same words on example, so the feature value is

$$(2 \times 8)/(11 + 12) = 0.59$$

5. A value in the range 0 to 1 that represents longest sub-sentence of T and H. The longest sub-sentence in given example is *Universitas Indonesia Salemba*, so the feature value is

$$(2 \times 3)/(11 + 12) = 0.26$$

6. If T and H begin with the same word, the value of the feature is 1, else 0. In the example, T starts with word *dia*, while H starts with word *beliau*. So, the feature value is 0.

Meanwhile, the second classifier runs on the bag-of-words features extracted from author's comment text. We shortlist top frequent phrases that occur on the candidate of T or H pairs. We test several supervised machine learning algorithms on second view of data.

3.4 Co-training

We conduct the experiment on 15,000 candidate pairs of T and H. The co-training algorithm used in our experiment is adjusted from original method introduced by [3]. Our algorithm is presented as follow.

```
Data: L for labeled data, U for unlabeled data
Take k random data from U as U';
set failed iteration 0;
while failed iteration < n do
    Train classifier h1 with first view of L;
    Train classifier h2 with second view of L;
    Classify U' with h1 obtaining U1';
    Classify U' with h2 obtaining U2';
    Assign label for best-classified examples in U1' and U2' and add them to L;
    if there is no example added to L then
        failed iteration++;
    else
        set failed iteration 0;
    end
    Take k data from U to replace the content of current U ';
end
```

Algorithm 1: Co-training

An unlabeled data is tagged a label if both classifiers agree on assigning the label with confidence level more than a specific threshold value (0.9 for first classifier, 0.5 for the second). Co-training ceases when the two classifiers are unable to agree on any of the same labels in n consecutive iterations. Another important parameter is k, a amount of unlabeled data to be retrieved per iteration. After conducting empirical experiment, the values of n and k are respectively set at 3 and 500.

4 Result and Evaluation

Before applying co-training algorithm, we test the performance of the classifiers on the seed data using the 10-fold cross-validation setting. LSTM model as first view classifier scores accuracy 58%. When the hand-crafted features augmented into LSTM, the accuracy of the classifier improves to 69%. After evaluating several traditional supervised machine learning approach, we choose Multinomial Naive Bayes as the classifier to train the model from second view of our data.

In the end, our study yields 1,857 sentence pairs that is iteratively labeled by co-training algorithm. The amount of data being labeled in each iteration is disparate, ranging from 0 to 58. We evaluate the result in 5 iterations (hereafter called the iteration group) by taking a random sample of 13 pairs for each label. The selected data is manually evaluated by human. The average accuracy on label classes E and NE are respectively 82% and 67%. Overall, the accuracy of classification is 76%. Figure 5 shows the sampling accuracy of each iteration group.

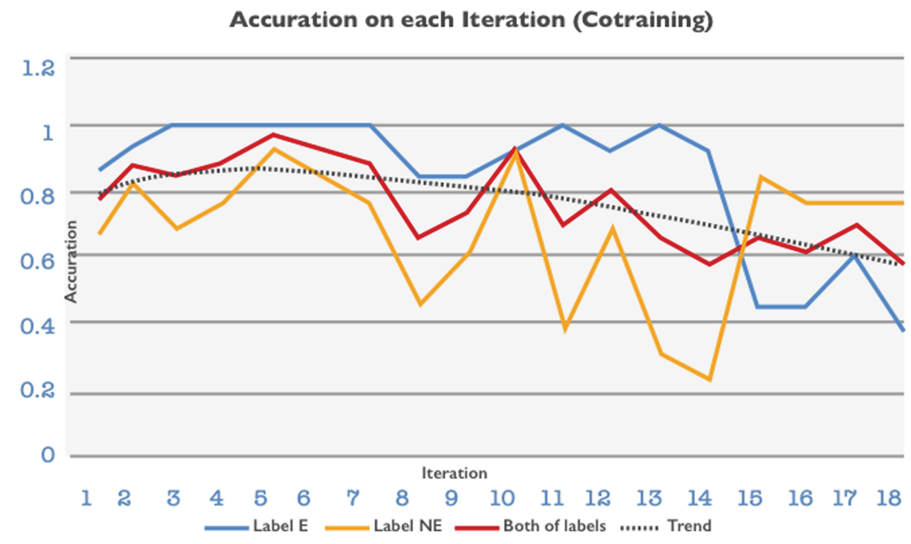


Fig. 5. Accuracy of iteration group on co-training

We observe the several cases in which co-training still make mistakes to assign label, i.e.:

- The difference between T and H is just the use of terms in foreign language (i.e. lexical substitution from English to Indonesian word),
- The difference between T and H is related to quantification problem, such as variation of number writing system (e.g. T uses term *5th*, while H uses term *fifth*). Another case is determining math inference between two text fragments.

From our experiment, we observe that most revision edits in Indonesian Wikipedia history data are generally paraphrases, or altering particular words with its synonyms. So, the instance of positive entailment relation is quite dominating.

5 Conclusion and Future Works

In this study, we apply a semi-supervised approach, using Co-training algorithm, to build an Indonesian Textual Entailment dataset from the scratch. We obtain data from Wikipedia revision history in Indonesian language through several stages of processing, namely: Wikipedia text extraction, T and H candidate formation, seed annotation, and feature engineering. The method requires data with two independent views. We extract the pairs of text before and after the revision (i.e. a pair of T and H) as first view and the author comment after revised the text as second view. By taking advantage of small portion of labeled data as seeds, co-training method automatically tags unlabeled data using two separate classifiers in each view. Wikipedia revision history is sufficient to produce considerably high-quality large dataset. Starting from only 400 annotated seeds, we can increase the size of the Indonesian Textual Entailment dataset by adding 1,857 new labeled data.

Although the performance of this initial study is good enough for pioneering of Indonesian Textual Entailment research, the further improvements are needed. The exploration on features engineering (especially syntactic and semantic features) is expected to gain the accuracy of classification. More careful inspection can be performed to learn better parameters for co-training algorithm. We also intend to test our methodology on other data sources, i.e. online news or social media.

Acknowledgement. We thank to Mirna Adriani for input and comments. This research was partially supported by PITTA UI Grant Contract No. 410/UN2.R3.1/HKP.05.00/2017. The first author was also partially funded by Bukalapak.

References

1. Alabbas, M.: A dataset for Arabic textual entailment. In: RANLP, pp. 7–13 (2013)
2. Androutsopoulos, I., Malakasiotis, P.: A survey of paraphrasing and textual entailment methods. J. Artif. Intell. Res. **38**(1), 135–187 (2010)
3. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: Proceedings of the 11th Annual Conference on Computational Learning Theory, pp. 92–100. Madison, WI (1998)
4. Bos, J., Markert, K.: Recognising textual entailment with logical inference. In: Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (2005)
5. Bos, J., Zanzotto, F.M., Pennacchiotti, M.: Textual entailment at evalita 2009. In: Proceedings of EVALITA 2009, pp. 1–7 (2009)

6. 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). Association for Computational Linguistics (2015)
7. Burger, J., Ferro, L.: Generating an entailment corpus from news headlines. In: Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment. EMSEE 2005, Stroudsburg, PA, USA, pp. 49–54. Association for Computational Linguistics (2005)
8. Clinchant, S., Goutte, C., Gaussier, E.: Lexical entailment for information retrieval. In: Lalmas, M., MacFarlane, A., Rüger, S., Tombros, A., Tsikrika, T., Yavlinsky, A. (eds.) ECIR 2006. LNCS, vol. 3936, pp. 217–228. Springer, Heidelberg (2006). https://doi.org/10.1007/11735106_20
9. 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
10. Ríos Gaona, M.A., Gelbukh, A., Bandyopadhyay, S.: Recognizing textual entailment using a machine learning approach. In: Sidorov, G., Hernández Aguirre, A., Reyes García, C.A. (eds.) MICAI 2010. LNCS (LNAI), vol. 6438, pp. 177–185. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-16773-7_15
11. Giampiccolo, D., Dang, H.T., Magnini, B., Dagan, I., Cabrio, E., Dolan, B.: The fourth PASCAL recognizing textual entailment challenge. In: Proceedings of the Fourth Text Analysis Conference, TAC 2008, Gaithersburg, Maryland, USA, November 17–19, 2008 (2008)
12. Giampiccolo, D., Magnini, B., Dagan, I., Dolan, B.: The third pascal recognizing textual entailment challenge. In: Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing. RTE 2007, Stroudsburg, PA, USA, pp. 1–9, Association for Computational Linguistics (2007)
13. Hickl, A., Williams, J., Bensley, J., Roberts, K., Rink, B., Shi, Y.: Recognizing textual entailment with LCC's groundhog system. In: Proceedings of the Second PASCAL Challenges Workshop, vol. 18 (2006)
14. Inkpen, D., Kipp, D., Nastase, V.: Machine learning experiments for textual entailment. In: Proceedings of the Second PASCAL Challenges Workshop on Recognizing Textual Entailment, pp. 10–15 (2006)
15. Kozareva, Z., Montoyo, A.: MLENT: the machine learning entailment system of the university of Alicante. In: Proceedings of the Second PASCAL Challenges Workshop on Recognizing Textual Entailment, pp. 10–15 (2006)
16. Landis, J., Koch, G.: The measurement of observer agreement for categorical data. Biometrics **33**, 159–174 (1977)
17. Lloret, E., Ferrández, Ó., Muñoz, R., Palomar, M.: A text summarization approach under the influence of textual entailment. In: NLPCS (2008)
18. Malakasiotis, P., Androutsopoulos, I.: Learning textual entailment using SVMs and string similarity measures. In: Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing. RTE 2007, Stroudsburg, PA, USA, pp. 42–47. Association for Computational Linguistics (2007)
19. Marelli, M., Menini, S., Baroni, M., Bentivogli, L., bernardi, R., Zamparelli, R.: A sick cure for the evaluation of compositional distributional semantic models. In: Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014). European Language Resources Association (ELRA) (2014)

20. Marzelou, E., Zourari, M., Giouli, V., Piperidis, S.: Building a greek corpus for textual entailment. In: LREC (2008)
21. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. In: Proceedings of the International Conference on Learning Representations (ICLR 2013) (2013). http://arxiv.org/abs/1301.3781
22. Negri, M., Kouylekov, M., Magnini, B.: Detecting expected answer relations through textual entailment. In: Gelbukh, A. (ed.) CICLing 2008. LNCS, vol. 4919, pp. 532–543. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78135-6_46
23. Padó, S., Galley, M., Jurafsky, D., Manning, C.: Robust machine translation evaluation with entailment features. In: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1, pp. 297–305. Association for Computational Linguistics (2009)
24. Peñas, A., Rodrigo, Á., Verdejo, F.: SPARTE, a test suite for recognising textual entailment in Spanish. In: Gelbukh, A. (ed.) CICLing 2006. LNCS, vol. 3878, pp. 275–286. Springer, Heidelberg (2006). https://doi.org/10.1007/11671299_29
25. Rocktäschel, T., Grefenstette, E., Hermann, K.M., Kociský, T., Blunsom, P.: Reasoning about entailment with neural attention. In: Proceedings of the International Conference on Learning Representations (ICLR 2016) (2016). http://arxiv.org/abs/1509.06664
26. 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, pp. 371–378, Association for Computational Linguistics (2005)
27. 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, pp. 1442–1451. Association for Computational Linguistics, June 2016
28. Zanzotto, F.M., Pennacchiotti, M.: Expanding textual entailment corpora from wikipedia using co-training. In: Proceedings of the 2nd Workshop on The People's Web Meets NLP: Collaboratively Constructed Semantic Resources. Coling 2010 Organizing Committee, Beijing, China, pp. 28–36, August 2010
29. Zanzotto, F.M., Pennacchiotti, M., Moschitti, A.: A machine learning approach to textual entailment recognition. Nat. Lang. Eng. **15**(4), 551–582 (2009)
30. Zeller, B., Padó, S.: A search task dataset for german textual entailment. In: Proceedings of the 10th International Conference on Computational Semantics (IWCS), pp. 288–299. Potsdam (2013)

Author Index

A. Gelbukh (Ed.): CICLing 2018, LNCS 13396, pp. 429–431, 2023.
https://doi.org/10.1007/978-3-031-23793-5