

Kôiti Hasida
Win Pa Pa (Eds.)

Communications in Computer and Information Science

781

Computational Linguistics

15th International Conference of the Pacific Association
for Computational Linguistics, PACLING 2017
Yangon, Myanmar, August 16–18, 2017
Revised Selected Papers

Communications in Computer and Information Science

781

Commenced Publication in 2007

Founding and Former Series Editors:

Alfredo Cuzzocrea, Xiaoyong Du, Orhun Kara, Ting Liu, Dominik Ślęzak,
and Xiaokang Yang

Editorial Board

Simone Diniz Junqueira Barbosa

*Pontifical Catholic University of Rio de Janeiro (PUC-Rio),
Rio de Janeiro, Brazil*

Phoebe Chen

La Trobe University, Melbourne, Australia

Joaquim Filipe

Polytechnic Institute of Setúbal, Setúbal, Portugal

Igor Kotenko

*St. Petersburg Institute for Informatics and Automation of the Russian
Academy of Sciences, St. Petersburg, Russia*

Krishna M. Sivalingam

Indian Institute of Technology Madras, Chennai, India

Takashi Washio

Osaka University, Osaka, Japan

Junsong Yuan

Nanyang Technological University, Singapore, Singapore

Lizhu Zhou

Tsinghua University, Beijing, China

More information about this series at <http://www.springer.com/series/7899>


Kôiti Hasida · Win Pa Pa (Eds.)

Computational Linguistics

15th International Conference of the Pacific Association
for Computational Linguistics, PACLING 2017
Yangon, Myanmar, August 16–18, 2017
Revised Selected Papers

Editors

Kôiti Hasida
Graduate School of Information Science
and Technology
The University of Tokyo
Tokyo
Japan

Win Pa Pa 
Natural Language Processing Lab
University of Computer Studies, Yangon
Yangon
Myanmar

ISSN 1865-0929 ISSN 1865-0937 (electronic)
Communications in Computer and Information Science
ISBN 978-981-10-8437-9 ISBN 978-981-10-8438-6 (eBook)
<https://doi.org/10.1007/978-981-10-8438-6>

Library of Congress Control Number: 2018935886

© Springer Nature Singapore Pte Ltd. 2018

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

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

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

Printed on acid-free paper

This Springer imprint is published by the registered company Springer Nature Singapore Pte Ltd.
part of Springer Nature
The registered company address is: 152 Beach Road, #21-01/04 Gateway East, Singapore 189721,
Singapore

Preface

This volume is a compilation of selected papers from PACLING 2017, which was the 15th in the series of conferences held since 1989. Most of these events were held in Australia, Japan, and Canada, but recently we have gathered in developing countries: Malaysia in 2011, Indonesia in 2015, and Myanmar this time.

This shift coincides with the prospect that these latter countries, among others, have bigger near-future potentials in realizing smart societies powered by the circulation of rich and abundant data. For instance, India is building a presence-less, paperless, and cashless service infrastructure consisting of a nationwide person authentication system, open-API private and public services, a national PDS (personal data store) for each individual to coordinate these services while utilizing their personal data. Cambodia also aims at a cashless society based on a still lighter infrastructure.

It is very likely that most other Asian countries share similar visions of near-future societies powered by data circulation. Such restructurings are not only inexpensive enough for developing countries, but also much easier in those countries than in advanced countries facing stronger opposition by many vested interests. I hence feel this year's PACLING in Myanmar, with a much higher literacy rate than in India and Cambodia, is exciting.

I hope the conference and the proceedings contribute to the construction of smart societies, as technologies to deal with language could address essential parts of the data circulation.

December 2017

Kôiti Hasida

Organization

Honorary Chair

Kôiti Hasida The University of Tokyo, Japan

Local Organizing Chair

Mie Mie Thet Thwin University of Computer Studies, Yangon, Myanmar

Local Organizing Committee

Nang Saing Moon Kham University of Computer Studies, Yangon, Myanmar
Win Pa Pa University of Computer Studies, Yangon, Myanmar

Program Committee

Kenji Araki Hokkaido University, Japan
Eiji Aramaki NAIST, Japan
Normaziah Abdul Aziz International Islamic University, Malaysia
Tetsuro Chino Toshiba Corporation, Japan
Khalid Choukri ELRA/ELDA, France
Koji Dosaka Akita Prefectural University, Japan
Alexander Gelbukh Instituto Politécnico Nacional (IPN), Mexico
Li Haizhou National University of Singapore, Singapore
Yoshihiko Hayashi Waseda University, Japan
Tin Myat Htwe KyaingTon Computer University, Myanmar
Bowen Hui University of British Columbia, Canada
Kentarô Inui Tohoku University, Japan
Kai Ishikawa NEC Corporation, Japan
Hiroyuki Kameda Tokyo University of Technology, Japan
Vlado Kesel Dalhousie University, Canada
Satoshi Kinoshita JAPIO, Japan
Kiyoshi Kogure Kanazawa Institute of Technology, Japan
Qin Lu The Hong Kong Polytechnic University, SAR China
Joseph Mariani LIMSI-CNRS, France
Robert Mercer The University of Western Ontario, Canada
Diego Mollá-Aliod Macquarie University, Australia
Hiromi Nakaiwa Nagoya University, Japan
Tin Htar New Magway Computer University, Myanmar
Fumihito Nishino Fujitsu Laboratories, Japan
Win Pa Pa University of Computer Studies, Yangon, Myanmar

Hamman Riza	Agency for the Assessment and Application of Technology (BPPT), Indonesia
Hiroaki Saito	Keio University, Japan
Kazutaka Shimada	Kyushu Institute of Technology, Japan
Akira Shimazu	Japan Advanced Institute of Science and Technology, Japan
Kiyooki Shirai	Japan Advanced Institute of Science and Technology, Japan
Khin Mar Soe	University of Computer Studies, Yangon, Myanmar
Virach Sornlertlamvanich	Thammasat University, Thailand
Thepchai Supnithi	NECTEC, Thailand
Hisami Suzuki	Microsoft, USA
Masami Suzuki	KDDI Research, Japan
Kumiko Tanaka	The University of Tokyo, Japan
Thanaruk Theeramunkong	Thammasat University, Thailand
Aye Thida	University of Computer Studies, Mandalay, Myanmar
Takenobu Tokunaga	Tokyo Institute of Technology, Japan
Mutsuko Tomokiyo	Université Grenoble Alpes, France
Yang Xiang	University of Guelph, Canada
Yuzana	Sittwe Computer University, Myanmar
Ingrid Zukerman	Monash University, Australia



Ministry of Education,
Myanmar



University of Computer
Studies, Yangon,
Myanmar

Contents

Semantics and Semantic Analysis

Detecting Earthquake Survivors with Serious Mental Affliction.	3
<i>Tatsuya Aoki, Katsumasa Yoshikawa, Tetsuya Nasukawa, Hiroya Takamura, and Manabu Okumura</i>	
A Deep Neural Architecture for Sentence-Level Sentiment Classification in Twitter Social Networking	15
<i>Huy Nguyen and Minh-Le Nguyen</i>	
Learning Word Embeddings for Aspect-Based Sentiment Analysis	28
<i>Duc-Hong Pham, Anh-Cuong Le, and Thi-Kim-Chung Le</i>	
BolLy: Annotation of Sentiment Polarity in Bollywood Lyrics Dataset	41
<i>G. Drushti Apoorva and Radhika Mamidi</i>	
Frame-Based Semantic Patterns for Relation Extraction	51
<i>Angrosh Mandya, Danushka Bollegala, Frans Coenen, and Katie Atkinson</i>	
Semantic Refinement GRU-Based Neural Language Generation for Spoken Dialogue Systems	63
<i>Van-Khanh Tran and Le-Minh Nguyen</i>	
Discovering Representative Space for Relational Similarity Measurement.	76
<i>Huda Hakami, Angrosh Mandya, and Danushka Bollegala</i>	
Norms of Valence and Arousal for 2,076 Chinese 4-Character Words	88
<i>Pingping Liu, Minglei Li, Qin Lu, and Buxin Han</i>	

Statistical Machine Translation

Integrating Specialized Bilingual Lexicons of Multiword Expressions for Domain Adaptation in Statistical Machine Translation	101
<i>Nasredine Semmar and Meriama Laib</i>	
Logical Parsing from Natural Language Based on a Neural Translation Model	115
<i>Liang Li, Yifan Liu, Zengchang Qin, Pengyu Li, and Tao Wan</i>	

Phrase-Level Grouping for Lexical Gap Resolution
in Korean-Vietnamese SMT 127
Seung Woo Cho, Eui-Hyeon Lee, and Jong-Hyeok Lee

Enhancing Pivot Translation Using Grammatical
and Morphological Information. 137
Hai-Long Trieu and Le-Minh Nguyen

Corpora and Corpus-Based Language Processing

Information-Structure Annotation of the “Balanced Corpus
of Contemporary Written Japanese”. 155
*Takuya Miyauchi, Masayuki Asahara, Natsuko Nakagawa,
and Sachi Kato*

Syntax and Syntactic Analysis

Khmer POS Tagging Using Conditional Random Fields 169
Sokunsatya Sangvat and Charnyote Pluempitiwiriyawej

Statistical Khmer Name Romanization 179
*Chenchen Ding, Vichet Chea, Masao Utiyama, Eiichiro Sumita,
Sethserey Sam, and Sopheap Seng*

Burmese (Myanmar) Name Romanization: A Sub-syllabic Segmentation
Scheme for Statistical Solutions 191
Chenchen Ding, Win Pa Pa, Masao Utiyama, and Eiichiro Sumita

Document Classification

Domain Adaptation for Document Classification by Alternately Using
Semi-supervised Learning and Feature Weighted Learning 205
Hiroyuki Shinnou, Kanako Komiya, and Minoru Sasaki

Information Extraction and Text Mining

End-to-End Recurrent Neural Network Models for Vietnamese
Named Entity Recognition: Word-Level Vs. Character-Level 219
Thai-Hoang Pham and Phuong Le-Hong

Nested Named Entity Recognition Using Multilayer Recurrent
Neural Networks. 233
Truong-Son Nguyen and Le-Minh Nguyen

Text Summarization

- Deletion-Based Sentence Compression Using Bi-enc-dec LSTM 249
Dac-Viet Lai, Nguyen Truong Son, and Nguyen Le Minh

Text and Message Understanding

- Myanmar Number Normalization for Text-to-Speech 263
Aye Mya Hlaing, Win Pa Pa, and Ye Kyaw Thu

- Expect the Unexpected: Harnessing Sentence Completion*
 for Sarcasm Detection 275
Aditya Joshi, Samarth Agrawal, Pushpak Bhattacharyya,
and Mark J. Carman

- Detecting Computer-Generated Text Using Fluency and Noise Features 288
Hoang-Quoc Nguyen-Son and Isao Echizen

Automatic Speech Recognition

- Speaker Adaptation on Myanmar Spontaneous Speech Recognition 303
Hay Mar Soe Naing and Win Pa Pa

- Exploring the Effect of Tones for Myanmar Language Speech Recognition
 Using Convolutional Neural Network (CNN) 314
Aye Nyein Mon, Win Pa Pa, and Ye Kyaw Thu

Spoken Language and Dialogue

- Listenability Measurement Based on Learners' Transcription Performance . . . 329
Katsunori Kotani and Takehiko Yoshimi

Speech Pathology

- A Robust Algorithm for Pathological-Speech Correction 341
Naim Terbeh and Mounir Zrigui

Speech Analysis

- Identification of Pronunciation Defects in Spoken Arabic Language 355
Naim Terbeh and Mounir Zrigui

- Author Index** 367

Semantics and Semantic Analysis



Detecting Earthquake Survivors with Serious Mental Affliction

Tatsuya Aoki¹(✉), Katsumasa Yoshikawa², Tetsuya Nasukawa²,
Hiroya Takamura¹, and Manabu Okumura¹

¹ Laboratory for Future Interdisciplinary Research of Science and Technology,
Tokyo Institute of Technology, Yokohama, Japan

{aoki,takamura,oku}@lr.pi.titech.ac.jp

² IBM Research - Tokyo, IBM Japan, Ltd., Tokyo, Japan

{KATSUY,NASUKAWA}@jp.ibm.com

Abstract. The 2011 Great East Japan Earthquake and 2016 Kumamoto earthquakes had a great impact on numerous people all over the world. In this paper, we focus on social media and the mental health of 2016 Kumamoto earthquake survivors. We first focus on the users who had experienced an earthquake and track their sentiments before and after the disaster using Twitter as a sensor. Consequently, we found that their emotional polarities switch from nervous during earthquakes and return to normal after huge earthquakes. However, we also found that some people did not go back to normal even after huge earthquakes subside. Against this background, we attempted to identify survivors who are suffering from serious mental distress concerning earthquakes. Our experimental results suggest that, besides the frequency of words related to earthquakes, the deviation in sentiment and lexical factors during the earthquake represent the mental conditions of Twitter users. We believe that the findings of this study will contribute to early mental health care for people suffering the aftereffects of a huge disaster.

1 Introduction

The Kumamoto earthquakes occurred on April 14th, 2016. Earthquakes of magnitude 7.0 occurred twice, and these huge shocks caused severe damage. As a result, these earthquakes damaged over 185,000 houses and 4,600 buildings, killed at least 50 people, and made approximately 180,000 people evacuees. Moreover, over 3,700 aftershocks occurred from April to June. The long-term aftershocks are a reason for the over 100 disaster-related deaths, including deaths indirectly related to the disaster such as death due to disaster-related stress. Additionally, because of the many aftershocks, some survivors suffered repeated post-quake trauma resulting in post-traumatic stress disorder.

In such an unusual condition, a microblog is a helpful tool to inform others of the current situation immediately. Many Twitter users tweeted about their safety or the emergency situation during the earthquakes. After the earthquakes, the survivors tended to use microblogs to contact their friends or people who

were in the same situation in order to share disaster-related information such as evacuation information and also to release anxiety by sending messages to each other. For earthquake survivors, information such as the status of the water supply is important to let people know their circumstances in a huge disaster. This kind of information is also important support because we are able to discover help messages based on the emergency messages from social media.

While many researchers have focused on retrieval of information regarding natural disasters [12, 14, 18], few have focused on the survivors' behaviors and their mental condition. Posts that exhibit signals of mental illness caused by unusual conditions increase during a disaster. The characteristics indicate that there are people in need of mental health care because of disaster-related mental aftereffects.

In this paper, we aim to identify people suffering severe mental stress due to earthquake. We also show the importance of social media to mental health by analyzing posts on Twitter of Kumamoto earthquake survivors. For constructing the dataset, we annotated the posts of Kumamoto survivors with earthquake-related stress levels judged by the content of their posts. Although there are people suffering from mental aftereffects of earthquakes, there has not been prior work on social media analysis that mentions mental health care for those people. It would be worthwhile to try to catch signals of mental illness from social media because we are not able to identify who needs mental health care after a large-scale disaster. We believe that this study will contribute to early mental health care for people suffering the aftereffects of a huge disaster.

Our main contributions in this study are as follows:

- We attempt to examine the relation between a huge earthquake and shifts in mental conditions by leveraging social media.
- We conduct a large-scale sentiment analysis using time series social media data. We track the sentiments of earthquake survivors before and after a huge disaster.
- We show the possibility that we can detect people who are suffering from serious mental stress concerning earthquakes on the basis of sentiment deviation and language use on social media.

2 Related Work

There is a wide range of research that uses microblogs as a sensor for detecting events [16], predicting geolocation [2], mining public opinion [13] and catching tendency [1, 10].

Vieweg et al. [19] suggested that microblogged information plays a role of a situational awareness observer when an emergency event occurs. There are also several research efforts that leverage microblogged information during natural disaster events for facilitating support [18]. Okazaki et al. [14] collected misinformation automatically in order to prevent false information from spreading through social media. Neubig et al. [12] studied ways of extracting safety information about people during the 2011 East Japan Earthquake from Twitter.

Table 1. Dataset for sentiment analysis

Period	March 1st, 2016–June 6th, 2016
#User	2,629
#Tweet	2,729,245

Varga et al. [18] built an information retrieval system to extract problem reports and aid messages from Twitter during natural disasters.

Recent studies have suggested that Twitter is a plentiful source of public health. Twitter allows us to access a large amount of information about public health including mental disorders. Previous studies connecting social media to mental health use knowledge from social data for predicting depression [4], measuring stress level [11], and exploring mental health and language [5, 7]

Nevertheless, there have been few investigations of the relation between real events and mental condition shifts which can be seen in social media. Choudhury et al. [3] studied the behavioral changes of new mothers and predicted significant postnatal behavioral changes based on prenatal information gathered on Twitter. Kumar et al. [9] focused on celebrity suicides that caused suicides of their fans and analyzed posting frequencies and content on Reddit after celebrity suicides by using a topic model.

3 Time Series Sentiment Analysis

As a preliminary experiment, we examined the sentiment of survivors before and after the Kumamoto earthquakes by constructing a sentiment model that measures the sentiment of users from their posts to a microblog. We used Twitter as a dataset and collected posts between March 1st, 2016 and June 6th, 2016. To focus on the Kumamoto earthquakes survivors, we analyzed only Twitter users whose location in the user profile or tweet location was Kumamoto. Table 1 summarizes our dataset.

3.1 Sentiment Model

First, we define the sentiment model and sentiment score. The sentiment model measures the user’s sentiment level. The output of the sentiment model is a sentiment score indicating the sentiment levels of users. We built the sentiment model by using tweets as input. Let \mathbf{D} be a set of tweets. $\mathbf{D}_{positive}$ and $\mathbf{D}_{negative} \subseteq \mathbf{D}$ are the sets of positive tweets and negative tweets respectively, which are subsets of \mathbf{D} . $\mathbf{D}_{polarity}$ is the set of tweets with positive/negative sentiment, and defined as $\mathbf{D}_{polarity} = \mathbf{D}_{positive} \cup \mathbf{D}_{negative}$. The sentiments of tweets are determined by the sentiment analyzer of IBM®Watson Explorer Advanced Edition Analytical Components V11.0 (WEX)¹. However, only a few tweets were provided

¹ IBM®Watson Explorer Advanced Edition Analytical Components V11.0 is a trademark in the United States and/or other countries of International Business Machines Corporation.

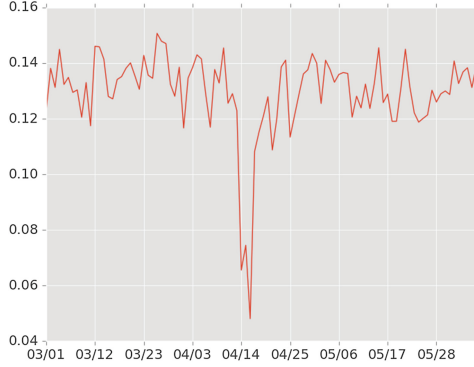


Fig. 1. Time Series Sentiment Score

with sentiments determined with the WEX sentiment analyzer because of errors regarding the dependency analysis and morphological analysis for short texts written in colloquial style. Thus, we used pattern matching with the Japanese Sentiment Dictionary [8, 17] in addition to the sentiment analyzer. If the results of the two methods contradict each other, we used both estimation results as the sentiment of the tweets. We thus permitted both positive and negative sentiments to be allocated to a tweet. The sentiment scoring function $s(\mathbf{D})$ is defined as:

$$s(\mathbf{D}) = g_1(\mathbf{D}_{positive}, \mathbf{D}_{negative}) \times g_2(\mathbf{D}, \mathbf{D}_{polarity}),$$

$$g_1(\mathbf{D}_{positive}, \mathbf{D}_{negative}) = \frac{|\mathbf{D}_{positive}| - |\mathbf{D}_{negative}|}{|\mathbf{D}_{positive}| + |\mathbf{D}_{negative}|},$$

$$g_2(\mathbf{D}, \mathbf{D}_{polarity}) = \frac{|\mathbf{D}_{polarity}|}{|\mathbf{D}|}.$$

The sentiment scoring function $s(\mathbf{D})$ consists of two factors, where $g_1(\mathbf{D}_{positive}, \mathbf{D}_{negative})$ returns indicates the inclination of tweets toward positive sentiment and $g_2(\mathbf{D}, \mathbf{D}_{polarity})$ returns the proportion of sentiment tweets.

3.2 Time Series Sentiment Score

For the dataset shown in Table 1, we demonstrate how the sentiment score changes day by day. We excluded tweets containing earthquake-related words², earthquake, aftershock, immediate report, shelter, shaking and goods, from $\mathbf{D}_{negative}$ in order to prevent the sentiment score from being biased toward negative due to those words; we would like to focus on the users' mental conditions.

² We extracted expressions which occur with high frequency and correlation by using WEX and defined them as earthquake-related words: 地震 (earthquake), 余震 (aftershock), 速報 (immediate report), 避難所 (shelter), 揺れ (shaking) and 物資 (goods).

Figure 1 shows the average daily sentiment score of the users in our dataset. As shown in Fig. 1, the sentiment score suddenly drops on April 14th, i.e., when the Kumamoto Earthquakes started, and then returns to the same tendency before the drop. The graph marks the lowest score (0.048pt) on April 16th, when the main shock of the Kumamoto Earthquakes occurred.

Table 2. Average sentiment scores by period

Period	Date	AVE. score	#Tweet
Before disaster	Mar. 1–Apr.13	0.134	1,016,068
During disaster	Apr. 13–Apr. 23	0.103	366,154
After disaster	Apr. 24–Jun. 6	0.131	1,347,023

Table 3. Average sentiment scores on different dataset

Group	AVE. score	#Tweet
Control	0.119	8,043
Schizophrenia	0.049	36,417
ADHD	0.042	29,454
Depression	0.029	38,782
PTSD	0.012	43,578
Borderline	0.004	29,648
Bipolar	-0.000	20,039

3.3 Comparison of Sentiment Score

To evaluate the sentiment scores before and after the earthquake, we split the dataset into three parts by date. We defined the 44 days before the first earthquake (April 14) as the *before-disaster period*, the 10 days after the first earthquake occurred as the *during-disaster period*, and the 44 days after the during-disaster period as the *after-disaster period*.

Table 2 summarizes the average sentiment scores within the whole period. As shown in Table 2, the score for the *during-disaster period* is 0.103pt and approximately 0.03pt lower than the scores of the *before- and after-disaster periods*. On the other hand, the scores are at the same level in the *before- and after-disaster periods*, which indicates that the sentiment scores of users returned to neutral after the huge earthquakes.

To investigate the relation between the mental health condition and the sentiment score, we built a new dataset to compare the scores on different groups: a control group and a mental-health-aware group. As the control group, we randomly picked users from Twitter and collected their most recent 150 tweets on October, 2016. As the mental health-aware group, we selected Twitter users who

Table 4. Example tweets of survivors after huge earthquakes

<p>地震、余震が続いてこれが震度4クラスが来るとなると緊張感で食事はおろか眠れなくなるが更に痩せてきてヒエロニスムのボッシュみたいな体型になりつつある。不健康極まりない</p> <p><i>I am under strain due to the continuing strong intensity of the earthquakes and aftershocks. I can't even sleep or eat anything.</i></p> <p><i>I have become thinner, like Hieronymus Bosch. My condition is really unhealthy.</i></p>
<p>お線香の味がする漢方薬と、抗不安薬を再開することになるなんて、本当に不本意。地震さえなければ...</p> <p><i>I resumed taking anti-anxiety drugs and Chinese medicine which tastes like an incense stick. I really hate them. I wish there were no earthquakes...</i></p>

Table 5. Dataset for serious survivor detection

Period	March 1st, 2016–June 6th, 2016
#User	289
#positive	29
#negative	260
#Tweet	366,154

had written the name of an illness in their Twitter profile and collected their most recent 150 tweets in October, 2016. We used the self diagnosis profile matching method that finds the users whose twitter profiles contain a name of a mental disorder. We defined the mental-health-aware group to be these users. Following the research of Coppersmith et al. [6], we focused on six mental health conditions: schizophrenia, attention deficit hyperactivity disorder (ADHD), depression, post-traumatic stress disorder (PTSD), borderline personality disorder (Borderline), and bipolar disorder (Bipolar). Next, we compared the sentiment score of the control group and those of other groups. For calculating the sentiment score, we regarded the tweets of the group as one large set of tweets.

Table 3 summarizes the average sentiment score of the control group and the six mental-health-aware groups. Compared with the score of the control group, the scores of the mental-health-aware groups tended to be lower. The average score of mental-health-aware groups was 0.023pt, close to the lowest score in Fig. 1 (0.048pt), i.e., the sentiment score on April 16th when the main shock of the Kumamoto Earthquakes occurred.

4 Seriously Affected Survivor Detection

4.1 Problem Setting

Our dataset included tweets from people suffering from aftereffects of the earthquake. Table 4 shows example tweets of survivors. Their mental condition might be negatively affected by their experiences in the earthquake. Such people would

require early mental care. Accordingly, we tried to detect users who exhibited signals of mental illnesses caused by the earthquake. To tackle this detection task, we built a new dataset that was a subset of the dataset used in Sect. 3 and randomly picked 300 users. We annotated each of the 300 users with one of two class labels: a user who does not care about disaster and a user who is severely affected by the disaster, on the basis of their mental health conditions determined from their tweets³. Since examining all tweets for each user is too costly for annotation purposes, we only examined the tweets containing earthquake-related words. We defined positive users as those whom both annotators labeled “affected” and negative users as otherwise. Table 5 shows the statistics of this dataset.

4.2 Methodology

Our goal for this task is to detect 29 positive users from 289 users. Since a large labeled dataset for this task is not available, we developed an unsupervised method, which relies on a final score indicating poor mental health of each user. The final score was calculated as the product of three factors described below.

Lexical Factor. It is supposed that users in poor mental health would have particular word usage characteristics. The lexical factor was calculated as the sum of three components: proportion of earthquake-related words, proportion of words associated with sleep disorders, and linguistic deviation of tweets.

– Proportion of Earthquake-related Words

Users concerned with the earthquake might be more likely to be mentally disordered. Thus, as a basic factor, we used the proportion of earthquake-related words $|\mathbf{D}_{eq-words}|/|\mathbf{D}|$, where $\mathbf{D}_{eq-words}$ is the set of tweets that contain earthquake-related words and \mathbf{D} is the set of all tweets of the user.

When the proportion of earthquake-related words is used together with other factors, we used its following normalized version for the purpose of keeping a good balance with others:

$$\frac{|\mathbf{D}_{eq-words}|}{|\mathbf{D}_{all-words}|} \times \frac{|\mathbf{D}_{eq-words}|}{|\mathbf{D}|},$$

where $\mathbf{D}_{all-words}$ is the set of tweets that contain earthquake-related words in the entire dataset.

– Proportion of Words Associated with Sleep-Disorder

Users in poor mental health often suffer from sleep disorders. We thus used $|\mathbf{D}_{sleep-disorder}|/|\mathbf{D}|$, where $\mathbf{D}_{sleep-disorder}$ is the set of tweets that contain

³ The actual annotation was a three class label: a user who does not care about the disaster, a user who is slightly concerned with the disaster, and user who is severely affected by the disaster. This annotation was conducted by two human annotators, and Cohen’s kappa for that is 0.75. By merging the first two classes into one class, we created an experimental dataset with two classes.

the words associated with sleep disorders. To obtain the tweets expressing sleep disorders, we collected expressions related to sleep disorders and added them to the dictionary of WEX.

– Emotion Bias

Users in poor mental health would more likely express anxiety and fear rather than optimistic emotions. We therefore used the following emotion deviation factor:

$$\frac{|\mathbf{D}_{anxiety}| + |\mathbf{D}_{fear}| - |\mathbf{D}_{optimistic}|}{|\mathbf{D}|},$$

where $\mathbf{D}_{anxiety}$, \mathbf{D}_{fear} , and $\mathbf{D}_{optimistic}$ are determined by a classifier for linguistic categories based on Linguistic Inquiry and Word Count [15].

User’s Vulnerability. We supposed that vulnerable users would take negative events more seriously. To implement this idea, we used Personality Insights⁴ to measure the vulnerability of each user. Personality Insights is a personality estimation service, which analyzes user personality in terms of psychology by feeding tweets into the system. We used its output to calculate the following factor:

$$Vulnerability(\mathbf{D}_{during}) + Vulnerability(\mathbf{D}_{after}) - Vulnerability(\mathbf{D}_{before}),$$

where \mathbf{D}_{before} , \mathbf{D}_{during} , and \mathbf{D}_{after} are respectively the sets of tweets posted in the periods of *before*, *during*, and *after the disaster*, as shown in Table 2.

Sentiment Deviation. We focused on the difference between the sentiment score of a user and the average score of all users. We additionally calculated earthquake-related sentiment scores with the same method as in Fig. 1. With the same method as in Fig. 1, we recalculated the sentiment scores for only tweets containing earthquake-related words; we called it the “earthquake-related sentiment score”. When calculating the sentiment scores of day k , we used the set of tweets posted from k to $k+10$ in order to alleviate any tweet sparseness of a user. We calculated two kinds of daily sentiment score. Then, we used these daily sentiment scores to calculate two deviation values. This factor is expressed as

$$\frac{1}{n} \sum_{i=1}^n (x_i - \bar{x}_i)^2 + \frac{1}{m} \sum_{j=1}^m (y_j - \bar{y}_j)^2,$$

where x_i is the sentiment score of the i -th day when the user posted a tweet and \bar{x}_i is the average sentiment score of day i in the entire dataset. n is the number of days when the user posted a tweet. Similarly, y_j is the earthquake-related sentiment score of the j -th day when the user posted a tweet containing earthquake-related words and \bar{y}_j is the average earthquake-related sentiment score of day j in the entire dataset. m is the number of days when the user posted a tweet that contains earthquake-related words.

⁴ <https://personality-insights-livedemo.mybluemix.net>.

4.3 Experiment

We evaluated our model using two measures. One was Recall@30 (R@30), which examined how many positive users are included in the top 30 users who have the highest final score. The other was average precision (AP), which investigated how much higher positive users are ranked compared with other users. Note that in the following ALL denotes the proposed method in which all factors were used.

Table 6. Results of Recall@30 and average precision

Feature	#detect user	R@30	AP
Proportion of Earthquake-related Words	10	0.35	0.32
+Vulnerability	11	0.38	0.37
+Vulnerability+Lexical Factor	14	0.48	0.41
+Vulnerability+Lexical Factor+Sentiment Deviation	17	0.59	0.45

Table 7. Ablation test of proposed model

Feature	#detect user	R@30	AP
ALL (Proposed)	17	0.59	0.45
w/o Proportion of EQ Words	16	0.55	0.45
w/o Vulnerability	16	0.55	0.46
w/o Lexical Factor	14	0.48	0.43
w/o Sentiment Deviation	14	0.48	0.41

Quantitative Evaluation. Table 6 shows that our model outperformed the other methods, achieving 0.59 pt in Recall@30 and 0.45 pt in AP.

Table 7 summarizes the results of the ablation test for the proposed model to determine which feature is effective for this task. First, we discuss the results of Recall@30. As presented in Table 7, the models without the proportion of earthquake-related words and user vulnerability features from the proposed model each performed 0.04pt worse than the proposed model. In addition, the models without sentiment deviation and lexical factor performed 0.11pt worse. Thus, the lexical factor and sentiment deviation have much more impact than the other two in our task.

Almost the same tendency as in Recall@30 can be seen on the results of average precision (AP). For the AP evaluation, the exclusion of lexical factors or sentiment deviations severely hurt performance. The only difference between Recall@30 and AP is that the model without the user’s vulnerability achieved the highest score at 0.46pt. However, we considered that the tiny improvement (0.01pt margin) is within the margin of error. Thus, these results indicate that the sentiment deviation and lexical factor are effective features for detecting positive users.

Table 8. Example tweets of positive user ranked seventh by the proposed model

今の地震ほんとに怖かった一余震でいつまで続くの
<i>The current earthquake was really scary. How long will the aftershocks last?</i>
なんかまた最近余震増えてない？もう夜中の地震すごい トラウマなんだけど
<i>Have the aftershocks increased recently? The earthquake at midnight was traumatic for me.</i>

Table 9. Example tweets of ordinary user ranked second by the proposed model

余震もうすぐ1.350回目 \ (^o^)/
<i>It is almost the 1350th aftershock.</i>
揺れない時間が増えて行く毎に心身共に安定してくる。
<i>The time period between tremors is increasing, My mind and body are becoming more stable.</i>

Qualitative Evaluation. Besides the quantitative evaluation, we focused on users labeled as “mental-health-aware” and checked the content of their tweets. Table 8 shows some posts by a user ranked by the proposed model at 7 out of 281. As shown in Table 8, the user’s posts describe his/her sleep disorder, and it can be seen that the user has negative feelings about the earthquakes. Moreover, as can be seen in the last sentence of Table 8 indicating that the earthquakes were a traumatic experience for him/her, this user was mentally affected by the earthquakes.

Error Analysis and Discussions. This section describes the error analysis and discusses the results. First, we report the negative users whom our model incorrectly detected as positive. Table 9 shows a user erroneously ranked in second place. Although posts related to earthquakes by this user increased after the huge earthquakes, there seemed to be no signal of mental illness.

Next, we report the positive users who were not detected by our model. Table 10 shows some examples of tweets of an undetected user who was the lowest-ranked positive user by the proposed model. Although the user is ranked

Table 10. Example tweets of positive user ranked 268th by the proposed model

もう揺れないでほしい。ほんとに。心臓いたい。
<i>Don't shake any more. My heart hurts.</i>
だめだほんと動悸と震えがとまらない ここ揺れてないのにほんとにやだ
<i>I can't stop shivering and throbbing. I feel bad even though there are no quakes here.</i>

268 by our model, it is highly probable that the user was mentally affected by the earthquakes. This is because our model was not able to capture the implicit expressions of the user. In order to capture the emotions of users from tweets, we should consider errors in preprocessing such as by using morphological analysis to process tweets that are usually written in a colloquial style.

Finally, we discuss the limitations of our work. One of the limitations is that our model does not take into account the temporal aspects of posting activities. For example, if the posting schedule suddenly changed after a huge disaster, it suggests there was some change in the user's life. Thus, if a user suddenly stops posting tweets, we would have to be careful with that user. Leveraging such temporal aspects would improve the ability of the method.

5 Conclusions

We aimed to detect the survivors who may suffer from mental illness due to the experience of a disaster. Our experimental results suggest that sentiment deviation and lexical factors including proportion of words associated with sleep disorders and emotion bias are effective features for detecting users who likely suffer from mental illness.

In the future, we will apply our method to mental health checks at school and companies by leveraging external information such as daily reports.

Acknowledgements. We are grateful to Koichi Kamijoh, Hiroshi Kanayama, Masayasu Muraoka, and the members of the IBM Research - Tokyo text mining team for their helpful discussions.

References

1. Aramaki, E., Maskawa, S., Morita, M.: Twitter catches the flu: detecting influenza epidemics using Twitter. In: Proceedings of EMNLP 2011, pp. 1568–1576 (2011)
2. Backstrom, L., Sun, E., Marlow, C.: Find me if you can: improving geographical prediction with social and spatial proximity. In: Proceedings of WWW 2010, pp. 61–70 (2010)
3. Choudhury, M.D., Counts, S., Horvitz, E.: Predicting postpartum changes in emotion and behavior via social media. In: Proceedings of CHI 2013, pp. 3267–3276 (2013)
4. Choudhury, M.D., Gamon, M., Counts, S., Horvitz, E.: Predicting depression via social media. In: Proceedings of ICWSM 2013 (2013)
5. Coppersmith, G., Dredze, M., Harman, C.: Quantifying mental health signals in Twitter. In: Proceedings of Workshop on Computational Linguistics and Clinical Psychology, pp. 51–60 (2014)
6. Coppersmith, G., Dredze, M., Harman, C., Hollingshead, K.: From ADHD to SAD: analyzing the language of mental health on Twitter through self-reported diagnoses. In: Proceedings of Workshop on Computational Linguistics and Clinical Psychology, pp. 1–10 (2015)

7. Gkotsis, G., Oellrich, A., Hubbard, T., Dobson, R., Liakata, M., Velupillai, S., Dutta, R.: The language of mental health problems in social media. In: Proceedings of Workshop on Computational Linguistics and Clinical Psychology, pp. 63–73 (2016)
8. Higashiyama, M., Inui, K., Matsumoto, Y.: Learning sentiment of nouns from selectional preferences of verbs and adjectives. In: Proceedings of Annual Meeting of the Association for Natural Language Processing (in Japanese), pp. 584–587 (2008)
9. Kumar, M., Dredze, M., Coppersmith, G., Choudhury, M.D.: Detecting changes in suicide content manifested in social media following celebrity suicides. In: Proceedings of HT 2015, pp. 85–94 (2015)
10. Lamb, A., Paul, M.J., Dredze, M.: Separating fact from fear: tracking flu infections on Twitter. In: Proceedings of NAACL:HLT 2013, pp. 789–795 (2013)
11. Lin, H., Jia, J., Nie, L., Shen, G., Chua, T.: What does social media say about your stress? In: Proceedings of IJCAI 2016, pp. 3775–3781 (2016)
12. Neubig, G., Matsubayashi, Y., Hagiwara, M., Murakami, K.: Safety information mining-what can NLP do in a disaster-. In: Proceedings of IJCNLP 2011, pp. 965–973 (2011)
13. O’Connor, B., Balasubramanyan, R., Routledge, B.R., Smith, N.A.: From tweets to polls: linking text sentiment to public opinion time series. In: Proceedings of ICWSM 2010, pp. 23–26 (2010)
14. Okazaki, N., Nabeshima, K., Watanabe, K., Mizuno, J., Inui, K.: Extracting and aggregating false information from microblogs. In: Proceedings of Workshop on Language Processing and Crisis Information, pp. 36–43 (2013)
15. Pennebaker, J.W., Francis, M.E., Booth, R.J.: Linguistic inquiry and word count: LIWC 2001. Lawrence Erlbaum Associates (2001)
16. Sakaki, T., Okazaki, M., Matsuo, Y.: Earthquake shakes Twitter users: real-time event detection by social sensors. In: Proceedings of WWW 2010, pp. 851–860 (2010)
17. Takamura, H., Inui, T., Okumura, M.: Extracting semantic orientations of words using spin model. In: Proceedings of ACL 2005, pp. 133–140 (2005)
18. Varga, I., Sano, M., Torisawa, K., Hashimoto, C., Ohtake, K., Kawai, T., Oh, J.H., De Saeger, S.: Aid is out there: looking for help from tweets during a large scale disaster. In: Proceedings of ACL 2013, pp. 1619–1629 (2013)
19. Vieweg, S., Hughes, A.L., Starbird, K., Palen, L.: Microblogging during two natural hazards events: what Twitter may contribute to situational awareness. In: Proceedings of CHI 2010, pp. 1079–1088 (2010)



A Deep Neural Architecture for Sentence-Level Sentiment Classification in Twitter Social Networking

Huy Nguyen^(✉) and Minh-Le Nguyen

Japan Advanced Institute of Science and Technology, Ishikawa, Japan
{huy.nguyen,nguyenml}@jaist.ac.jp

Abstract. This paper introduces a novel deep learning framework including a lexicon-based approach for sentence-level prediction of sentiment label distribution. We propose to first apply semantic rules and then use a Deep Convolutional Neural Network (DeepCNN) for character-level embeddings in order to increase information for word-level embedding. After that, a Bidirectional Long Short-Term Memory network (BiLSTM) produces a sentence-wide feature representation from the word-level embedding. We evaluate our approach on three twitter sentiment classification datasets. Experimental results show that our model can improve the classification accuracy of sentence-level sentiment analysis in Twitter social networking.

Keywords: Twitter · Sentiment classification · Deep learning

1 Introduction

Twitter sentiment classification have intensively researched in recent years [6, 14]. Different approaches were developed for Twitter sentiment classification by using machine learning such as Support Vector Machine (SVM) with rule-based features [15] and the combination of SVMs and Naive Bayes (NB) [17]. In addition, hybrid approaches combining lexicon-based and machine learning methods also achieved high performance described in [13]. However, a problem of traditional machine learning is how to define a feature extractor for a specific domain in order to extract important features.

Deep learning models are different from traditional machine learning methods in that a deep learning model does not depend on feature extractors because features are extracted during training progress. The use of deep learning methods becomes to achieve remarkable results for sentiment analysis [4, 10, 20]. Some researchers used Convolutional Neural Network (CNN) for sentiment classification. CNN models have been shown to be effective for NLP. For example, [10] proposed various kinds of CNN to learn sentiment-bearing sentence vectors, [4] adopted two CNNs in character-level to sentence-level representation

for sentiment analysis. [20] constructs experiments on a character-level CNN for several large-scale datasets. In addition, Long Short-Term Memory (LSTM) is another state-of-the-art semantic composition model for sentiment classification with many variants described in [5]. The studies reveal that using a CNN is useful in extracting information and finding feature detectors from texts. In addition, a LSTM can be good in maintaining word order and the context of words. However, in some important aspects, the use of CNN or LSTM separately may not capture enough information.

Inspired by the models above, the goal of this research is using a Deep Convolutional Neural Network (DeepCNN) to exploit the information of characters of words in order to support word-level embedding. A Bi-LSTM produces a sentence-wide feature representation based on these embeddings. The Bi-LSTM is a version of [7] with Full Gradient described in [8]. In addition, the rules-based approach also effects classification accuracy by focusing on important sub-sentences expressing the main sentiment of a tweet while removing unnecessary parts of a tweet. The paper makes the following contributions:

- We construct a tweet processor removing unnecessary sub-sentences from tweets in order that the model learns important information in a tweet. We share ideas with [1, 6], however, our tweet processor keeps emoticons in tweets and only uses rules to remove non-essential parts for handling negation.
- We train DeepCNN with Wide convolution on top of character embeddings to produce feature maps which capture the morphological and shape information of a word. The morphological and shape information illustrate how words are formed, and their relationship to other words. DeepCNN transforms the character-level embeddings into global fixed-sized feature vectors at higher abstract level. Such character feature vectors contribute enriching the information of words in a sentence.
- We create an integration of global fixed-size character feature vectors and word-level embedding for the Bi-LSTM. The Bi-LSTM connects the information of words in a sequence and maintains the order of words for sentence-level representation.

The organization of the present paper is as follows: In Sect. 2, we describe the model architecture which introduces the structure of the model. We explain the basic idea of model and the way of constructing the model. Section 3 shows results and analysis and Sect. 4 summarizes this paper.

2 Model Architecture

2.1 Basic Idea

Our proposed model consists in a deep learning classifier and a tweet processor. The deep learning classifier is a combination of DeepCNN and Bi-LSTM. The tweet processor standardizes tweets and then applies rules called semantic rules on datasets. We construct a framework that treats the deep learning classifier and

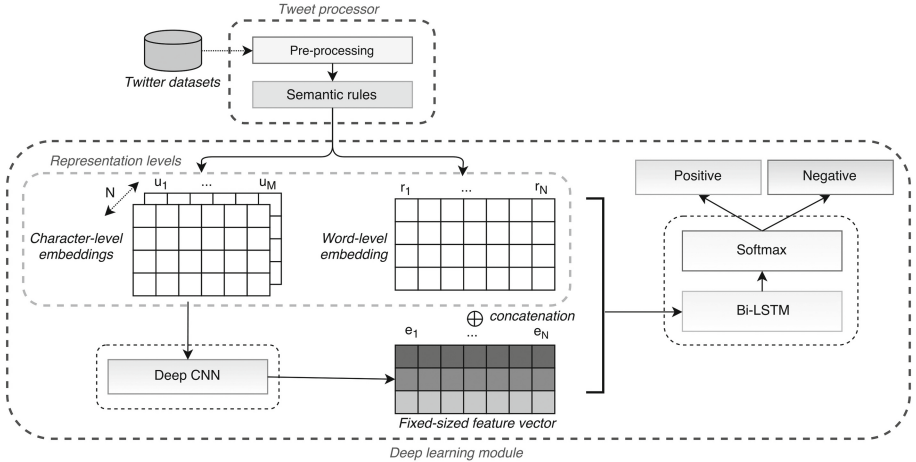


Fig. 1. The overview of a deep learning system.

the tweet processor as two distinct components. We believe that standardizing data is an important step to achieve high classification accuracy. To formulate our problem in increasing the accuracy of the classifier, we illustrate our model in Fig. 1 as follows:

1. Tweets are firstly considered via a processor based on preprocessing steps [6] and the semantic rules-based method [1] in order to standardize tweets and capture only important information containing the main sentiment of a tweet.
2. We use DeepCNN with Wide convolution for character-level embeddings. A wide convolution can learn to recognize specific n -grams at every position in a word that allows features to be extracted independently of these positions in the word. These features maintain the order and relative positions of characters. A DeepCNN is constructed by two wide convolution layers and the need of multiple wide convolution layers is widely accepted that a model constructing by multiple processing layers have the ability to learn representations of data with higher levels of abstraction [11]. Therefore, we use DeepCNN for character-level embeddings to support morphological and shape information for a word. The DeepCNN produces N global fixed-sized feature vectors for N words.
3. A combination of the global fixed-size feature vectors and word-level embedding is fed into Bi-LSTM. The Bi-LSTM produces a sentence-level representation by maintaining the order of words.

Our work is philosophically similar to [4]. However, our model is distinguished with their approaches in two aspects:

- Using DeepCNN with two wide convolution layers to increase representation with multiple levels of abstraction.

- Integrating global character fixed-sized feature vectors with word-level embedding to extract a sentence-wide feature set via Bi-LSTM. This deals with three main problems: (i) Sentences have any different size; (ii) The semantic and the syntactic of words in a sentence are captured in order to increase information for a word; (iii) Important information of characters that can appear at any position in a word are extracted.

In sub-section B, we introduce various kinds of dataset. The modules of our model are constructed in other sub-sections.

2.2 Data Preparation

- *Stanford - Twitter Sentiment Corpus (STS Corpus)*: STS Corpus contains 1,600K training tweets collected by a crawler from [6]. [6] constructed a test set manually with 177 negative and 182 positive tweets. The Stanford test set is small. However, it has been widely used in different evaluation tasks [2, 4, 6].
- *Sanders - Twitter Sentiment Corpus*: This dataset consists of hand-classified tweets collected by using search terms: *#apple*, *#google*, *#microsoft* and *#twitter*. We construct the dataset as [3] for binary classification.
- *Health Care Reform (HCR)*: This dataset was constructed by crawling tweets containing the hashtag *#hcr* [16]. Task is to predict positive/negative tweets [3].

2.3 Preprocessing

We firstly take unique properties of Twitter in order to reduce the feature space such as *Username*, *Usage of links*, *None*, *URLs* and *Repeated Letters*. We then process *retweets*, *stop words*, *links*, *URLs*, *mentions*, *punctuation* and *accentuation*. For emoticons, [6] revealed that the training process makes the use of emoticons as noisy labels and they stripped the emoticons out from their training dataset because [6] believed that if we consider the emoticons, there is a negative impact on the accuracies of classifiers. In addition, removing emoticons makes the classifiers learns from other features (e.g. unigrams and bi-grams) presented in tweets and the classifiers only use these non-emoticon features to predict the sentiment of tweets. However, there is a problem is that if the test set contains emoticons, they do not influence the classifiers because emoticon features do not contain in its training data. This is a limitation of [6], because the emoticon features would be useful when using deep learning for classifying the test data. Deep learning model captures emoticons as features and models syntactic context information for words. Therefore, we keep emoticon features in the datasets because deep learning models can capture more information from emoticon features for increasing classification accuracy.

Table 1. Semantic rules [1]

Rule	Semantic rules	Example - STS corpus	Output
R11	If a sentence contains “but”, disregard all previous sentiment and only take the sentiment of the part after “but”	@kirstiealley my dentist is great <i>but</i> she’s expensive...=(she’s expensive...=(
R12	If a sentence contains “despite”, only take sentiment of the part before “despite”	I’m not dead <i>despite</i> rumours to the contrary	I’m not dead
R13	If a sentence contains “unless”, and “unless” is followed by a negative clause, disregard the “unless” clause	laptop charger is broken - <i>unless</i> a little cricket set up home inside it overnight. Typical at the worst possible time	laptop charger is broken
R14	If a sentence contains “while”, disregard the sentence following the “while” and take the sentiment only of the sentence that follows the one after the “while”	My throat is killing me, and <i>While</i> I got a decent night’s sleep last night, I still feel like I’m about to fall over	I still feel like I’m about to fall over
R15	If the sentence contains “however”, disregard the sentence preceding the “however” and take the sentiment only of the sentence that follows the “however”	@lonedog bwahahah...you are amazing! <i>However</i> , it was quite the letdown	it was quite the letdown

2.4 Semantic Rules (SR)

In Twitter social networking, people express their opinions containing sub-sentences. These sub-sentences using specific PoS particles (Conjunction and Conjunctive adverbs), like “*but, while, however, despite, however*” have different polarities. However, the overall sentiment of tweets often focus on certain sub-sentences. For example:

- @lonedog *bwahahah...you are amazing! However, it was quite the letdown.*
- @kirstiealley *my dentist is great but she’s expensive...=(*

In two tweets above, the overall sentiment is negative. However, the main sentiment is only in the sub-sentences following *but* and *however*. This inspires a processing step to remove unessential parts in a tweet. Rule-based approach can assist these problems in handling negation and dealing with specific PoS particles led to effectively affect the final output of classification [1, 18]. [1] summarized a full presentation of their semantic rules approach and devised ten semantic rules in their hybrid approach based on the presentation of [18]. We use five rules in the semantic rules set because other five rules are only used to compute polarity of words after POS tagging or Parsing steps. We follow the same naming convention for rules utilized by [1] to represent the rules utilized in

our proposed method. The rules utilized in the proposed method are displayed in Table 1 in which is included examples from STS Corpus and output after using the rules. Table 2 illustrates the number of processed sentences on each dataset.

Table 2. The number of tweets are processed by semantic rules

Dataset	Set	# Sentences/tweets
STS corpus	Train	138703
	Test	25
Sanders	Train	39
	Dev	74
	Test	54
HCR	Train	164

2.5 Representation Levels

To construct embedding inputs for our model, we use a fixed-sized word vocabulary V^{word} and a fixed-sized character vocabulary V^{char} . Given a word w_i is composed from characters $\{c_1, c_2, \dots, c_M\}$, the character-level embeddings are encoded by column vectors u_i in the embedding matrix $W^{char} \in \mathbb{R}^{d^{char} \times |V^{char}|}$, where V^{char} is the size of the character vocabulary. For word-level embedding r_{word} , we use a pre-trained word-level embedding with dimension 200 or 300. A pre-trained word-level embedding can capture the syntactic and semantic information of words [12]. We build every word w_i into an embedding $v_i = [r_i; e_i]$ which is constructed by two sub-vectors: the word-level embedding $r_i \in \mathbb{R}^{d^{word}}$ and the character fixed-size feature vector $e_i \in \mathbb{R}^l$ of w_i where l is the length of the filter of convolutions. We have N character fixed-size feature vectors corresponding to word-level embedding in a sentence.

2.6 Deep Learning Module

DeepCNN in the deep learning module is illustrated in Fig. 2. The DeepCNN has two wide convolution layers. The first layer extracts local features around each character windows of the given word and using a max pooling over character windows to produce a global fixed-sized feature vector for the word. The second layer retrieves important context characters and transforms the representation at previous level into a representation at higher abstract level. We have N global character fixed-sized feature vectors for N words.

In the next step of Fig. 1, we construct the vector $v_i = [r_i, e_i]$ by concatenating the word-level embedding with the global character fixed-size feature vectors. The input of Bi-LSTM is a sequence of embeddings $\{v_1, v_2, \dots, v_N\}$. The use of the global character fixed-size feature vectors increases the relationship of words in the word-level embedding. The purpose of this Bi-LSTM is to capture the context of words in a sentence and maintain the order of words toward to extract

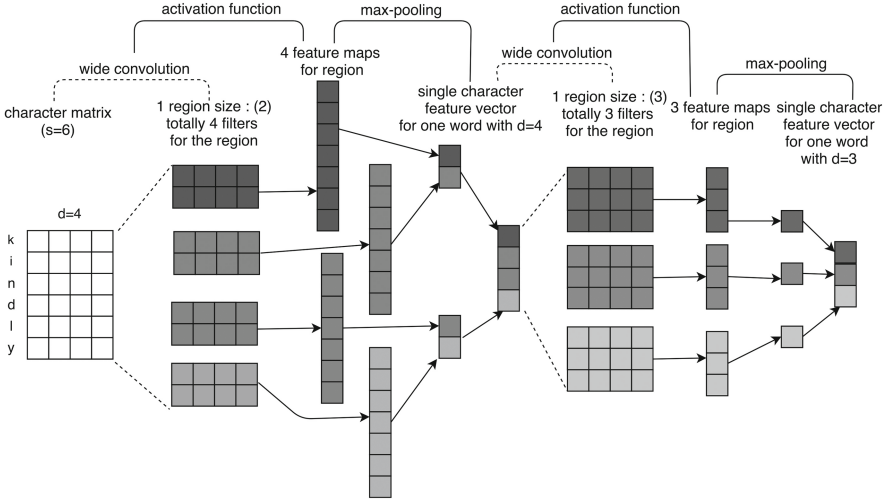


Fig. 2. Deep Convolutional Neural Network (DeepCNN) for the sequence of character embeddings of a word. For example with 1 region size is 2 and 4 feature maps in the first convolution and 1 region size is 3 with 3 feature maps in the second convolution.

a sentence-level representation. The top of the model is a softmax function to predict sentiment label. We describe in detail the kinds of CNN and LSTM that we use in the next sub-part 1 and 2.

Convolutional Neural Network. The one-dimensional convolution called time-delay neural net has a filter vector m and take the dot product of filter m with each m -grams in the sequence of characters $s_i \in R$ of a word in order to obtain a sequence c :

$$c_j = m^T s_{j-m+1:j} \quad (1)$$

Based on Eq. 1, we have two types of convolutions that depend on the range of the index j . The narrow type of convolution requires that $s \geq m$ and produce a sequence $c \in R^{s-m+1}$. The wide type of convolution does not require on s or m and produce a sequence $c \in R^{s+m-1}$. Out-of-range input values s_i where $i < 1$ or $i > s$ are taken to be zero. We use wide convolution for our model.

Wide Convolution. Given a word w_i composed of M characters $\{c_1, c_2, \dots, c_M\}$, we take a character embedding $u_i \in R^d$ for each character c_i and construct a character matrix $W^{char} \in R^{d \times |V^{chr}|}$ as following Eq. 2:

$$W^{char} = \begin{bmatrix} | & | & | & | \\ u_1 & \dots & u_M \\ | & | & | & | \end{bmatrix} \quad (2)$$

The values of the embeddings u_i are parameters that are optimized during training. The trained weights in the filter m correspond to a feature detector which

learns to recognize a specific class of n -grams. The n -grams have size $n \geq m$. The use of a wide convolution has some advantages more than a narrow convolution because a wide convolution ensures that all weights of filter reach the whole characters of a word at the margins. The resulting matrix has dimension $d \times (s + m - 1)$.

Long Short-Term Memory. Long Short-Term Memory networks usually called LSTMs are a improved version of RNN. The core idea behind LSTMs is the cell state which can maintain its state over time, and non-linear gating units which regulate the information flow into and out of the cell. The LSTM architecture that we used in our proposed model is described in [7]. A single LSTM memory cell is implemented by the following composite function:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \quad (3)$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \quad (4)$$

$$c_t = f_t c_{t-1} + i_t \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \quad (5)$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o) \quad (6)$$

$$h_t = o_t \tanh(c_t) \quad (7)$$

where σ is the logistic sigmoid function, i, f, o and c are the *input gate*, *forget gate*, *output gate*, *cell* and *cell input* activation vectors respectively. All of them have a same size as the hidden vector h . W_{hi} is the hidden-input gate matrix, W_{xo} is the input-output gate matrix. The bias terms which are added to i, f, c and o have been omitted for clarity. In addition, we also use the full gradient for calculating with full backpropagation through time (BPTT) described in [8]. A LSTM gradients using finite differences could be checked and making practical implementations more reliable.

2.7 Regularization

For regularization, we use a constraint on l_2 - norms of the weight vectors [9].

3 Results and Analysis

3.1 Experimental setups

For the Stanford Twitter Sentiment Corpus (STS Corpus), we use the number of samples as [4]. The training data is selected 80K tweets for a training data and 16K tweets for a development set randomly from the training data of [6]. We conduct a binary prediction for STS Corpus.

For Sander dataset, we use standard 10-fold cross validation as [3]. We construct a development set by selecting 10% randomly from 9-fold training data.

In Health Care Reform Corpus, we also select 10% randomly for a development set in a training set and construct as [3] for comparison. We describe the summary of datasets in Table 3.

Table 3. Summary statistics for the datasets after using semantic rules. c : the number of classes. N : The number of tweets. l_w : Maximum sentence length. l_c : Maximum character length. $|V_w|$: Word alphabet size. $|V_c|$: Character alphabet size.

Data	Set	N	c	l_w	l_c	$ V_w $	$ V_c $	$Test$
STS	Train	80 K	2	33	110	67083	134	-
	Dev	16 K		28	48			
	Test	359		21	16			
Sanders	Train	991	2	31	33	3379	84	CV
	Dev	110		27	47			
	Test	122		28	21			
HCR	Train	621	2	25	70	3100	60	-
	Dev	636		26	16			
	Test	665		20	16			

Hyperparameters. For all datasets, the filter window size (h) is 7 with 6 feature maps each for the first convolution layer, the second convolution layer has a filter window size of 5 with 14 feature maps each. Dropout rate (p) is 0.5, l_2 constraint, learning rate is 0.1 and momentum of 0.9. Mini-batch size for STS Corpus is 100 and others are 4. In addition, training is done through stochastic gradient descent over shuffled mini-batches with Adadelta update rule [19].

Pre-trained Word Vectors. We use the publicly available *Word2Vec*¹ trained from 100 billion words from Google and *TwitterGlove*² of Stanford is performed on aggregated global word-word co-occurrence statistics from a corpus. *Word2Vec* has dimensionality of 300 and *Twitter Glove* have dimensionality of 200. Words that do not present in the set of pre-train words are initialized randomly.

3.2 Experimental Results

Table 4 shows the results of our model for sentiment classification against other models. We compare our model performance with the approaches of [4, 6] on STS Corpus. [6] reported the results of Maximum Entropy (MaxEnt), Naive Bayes (NB), SVM on STS Corpus having good performances in previous time. The model of [4] is a state-of-the-art so far by using a CharSCNN. As can be seen, 86.63 is the best prediction accuracy of our model so far for the STS Corpus.

For Sanders and HCR datasets, we compare results with the model of [3] that used an ensemble of multiple base classifiers (ENS) such as NB, Random Forest (RF), SVM and Logistic Regression (LR). The ENS model is combined with bag-of-words (BoW), feature hashing (FH) and lexicons (lex). The model of [3] is a state-of-the-art on Sanders and HCR datasets. Our models outperform the model of [3] for the Sanders dataset and HCR dataset.

¹ <https://code.google.com/p/word2vec/>.

² <https://nlp.stanford.edu/projects/glove/>.

Table 4. Accuracy of different models for binary classification

Model	STS	Sanders	HCR
CharSCNN/pre-training [4]	86.4	-	-
CharSCNN/random [4]	81.9	-	-
SCNN/pre-training [4]	85.2	-	-
SCNN/random [4]	82.2	-	-
MaxEnt [6]	83.0	-	-
NB [6]	82.7	-	-
SVM [6]	82.2	-	-
SVM-BoW	-	82.43	73.99
SVM-BoW + lex	-	83.98	75.94
RF-BoW	-	79.24	70.83
RF-BoW + lex	-	82.35	72.93
LR-BoW	-	77.45	73.83
LR-BoW + lex	-	79.49	74.73
MNB-BoW	-	79.82	72.48
MNB-BoW + lex	-	83.41	75.33
ENS (RF + MNB + LR) - BoW	-	-	75.19
ENS (SVM + RF + MNB + LR) - BoW + lex	-	-	76.99
ENS (SVM + RF + MNB + LR) - BoW	-	82.76	-
ENS (SVM + RF + MNB) - BoW + lex	-	84.89	-
DeepCNN + SR + Glove	85.23	62.38	76.84
Bi-LSTM + SR + Glove	85.79	84.32	78.49
(DeepCNN + Bi-LSTM) + SR + Glove	86.63	85.14	79.55
(DeepCNN + Bi-LSTM) + SR + GoogleW2V	86.35	85.05	80.9
(DeepCNN + Bi-LSTM) + GoogleW2V	86.07	84.23	80.75

3.3 Analysis

As can be seen, the models with SR outperforms the model with no SR. Semantic rules are effective in order to increase classification accuracy. Table 5 describes the comparison between the model using SR and the model without SR. In Table 5, we list examples with its true labels that our model without SR fails in prediction and the red detectors affect the model to fail in prediction. The semantic rules assist our model to succeed in predicting true labels. We also conduct two experiments on two separate models: DeepCNN and Bi-LSTM in order to show the effectiveness of combination of DeepCNN and Bi-LSTM. In addition, the model using *TwitterGlove* outperform the model using *GoogleW2V* because *TwitterGlove* captures more information in Twitter than *GoogleW2V*. These results show that the character-level information and SR have a great

Table 5. The label prediction between the model using Semantic rules and the model without Semantic rules

Model	Inputs from HCR dataset	Label	Predict
(DeepCNN + Bi-LSTM) without SR	#hcr #whitehouse #Obama - To stay alive in this country, isn't war in another country, but war right here to live without proper care	Positive	False
	I'm not necessarily against #hcr, but i feel ignored as a late-twenties, single, childless woman above the poverty line & well under wealthy	Negative	False
	#Stupak to vote "Yes" on #HCR - but in exchange for what? #underthebus	Negative	False
(DeepCNN + Bi-LSTM) using SR	War right here to live without proper care	Positive	True
	I feel ignored as a late-twenties, single, childless woman above the poverty line & well under wealthy	Negative	True
	In exchange for what? #underthebus	Negative	True

impact on Twitter Data. The pre-train word vectors are good, universal feature extractors. The difference between our model and other approaches is the ability of our model to capture important features by using SR and combine these features at high benefit. The use of DeepCNN can learn a representation of words in higher abstract level. The combination of global character fixed-sized feature vectors and a word embedding helps the model to find important detectors for particles such as 'not' that negate sentiment and potentiate sentiment such as 'too', 'so' standing beside expected features. The model not only learns to recognize single n-grams, but also patterns in n-grams lead to form a structure significance of a sentence.

4 Conclusions

In the present work, we have pointed out that the use of character embeddings through a DeepCNN to enhance information for word embeddings built on top of *Word2Vec* or *TwitterGlove* improves classification accuracy in Tweet sentiment classification. Our results add to the well-establish evidence that character vectors are an important ingredient for word-level in deep learning for NLP. In addition, semantic rules contribute handling non-essential sub-tweets in order to improve classification accuracy.

Acknowledgment. This work was supported by the JSPS KAKENHI Grant number JP15K16048.

References

1. Appel, O., Chiclana, F., Carter, J., Fujita, H.: A hybrid approach to the sentiment analysis problem at the sentence level. *Knowl. Based Syst.* **108**, 110–124 (2016)
2. Bravo-Marquez, F., Mendoza, M., Poblete, B.: Combining strengths, emotions and polarities for boosting twitter sentiment analysis. In: *Proceedings of the Second International Workshop on Issues of Sentiment Discovery and Opinion Mining*, p. 2. ACM (2013)
3. Da Silva, N.F.F., Hruschka, E.R., Hruschka, E.R.: Tweet sentiment analysis with classifier ensembles. *Decis. Support Syst.* **66**, 170–179 (2014)
4. Dos Santos, C.N., Gatti, M.: Deep convolutional neural networks for sentiment analysis of short texts. In: *COLING*, pp. 69–78 (2014)
5. Gers, F.A., Schmidhuber, J.: Recurrent nets that time and count. In: *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks, IJCNN 2000*, vol. 3, pp. 189–194. IEEE (2000)
6. Go, A., Bhayani, R., Huang, L.: Twitter sentiment classification using distant supervision. *CS224N Project Report*, Stanford, **1**(12) (2009)
7. Graves, A.: Generating sequences with recurrent neural networks. *arXiv preprint [arXiv:1308.0850](https://arxiv.org/abs/1308.0850)* (2013)
8. Graves, A., Schmidhuber, J.: Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Netw.* **18**(5), 602–610 (2005)
9. Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.R.: Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint [arXiv:1207.0580](https://arxiv.org/abs/1207.0580)* (2012)
10. Kim, Y.: Convolutional neural networks for sentence classification. *CoRR*, abs/1408.5882 (2014)
11. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. *Nature* **521**(7553), 436–444 (2015)
12. 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*, vol. 26, pp. 3111–3119. Curran Associates Inc. (2013)
13. Muhammad, A., Wiratunga, N., Lothian, R.: Contextual sentiment analysis for social media genres. *Knowl. Based Syst.* **108**, 92–101 (2016)
14. Nakov, P., Ritter, A., Rosenthal, S., Sebastiani, F., Stoyanov, V.: SemEval-2016 task 4: sentiment analysis in twitter. In: *Proceedings of SemEval*, pp. 1–18 (2016)
15. Silva, J., Coheur, L., Mendes, A.C., Wichert, A.: From symbolic to sub-symbolic information in question classification. *Artif. Intell. Rev.* **35**(2), 137–154 (2011)
16. Speriosu, M., Sudan, N., Upadhyay, S., Baldridge, J.: Twitter polarity classification with label propagation over lexical links and the follower graph. In: *Proceedings of the First workshop on Unsupervised Learning in NLP*, pp. 53–63. Association for Computational Linguistics (2011)
17. 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*, vol. 2, pp. 90–94. Association for Computational Linguistics (2012)
18. Xie, Y., Chen, Z., Kumpeng Zhang, Y., Cheng, D.K., Honbo, A.A., Choudhary, A.N.: MuSES: multilingual sentiment elicitation system for social media data. *IEEE Intell. Syst.* **29**(4), 34–42 (2014)

19. Zeiler, M.D.: ADADELTA: an adaptive learning rate method. arXiv preprint [arXiv:1212.5701](https://arxiv.org/abs/1212.5701) (2012)
20. Zhang, X., LeCun, Y.: Text understanding from scratch. CoRR, abs/1502.01710 (2015)



Learning Word Embeddings for Aspect-Based Sentiment Analysis

Duc-Hong Pham^{1,3}, Anh-Cuong Le²(✉) , and Thi-Kim-Chung Le⁴

¹ Faculty of Information Technology, University of Engineering and Technology,
Vietnam National University, Hanoi, Vietnam

² NLP-KD Lab, Faculty of Information Technology, Ton Duc Thang University,
Ho Chi Minh City, Vietnam
leanhcuong@tdt.edu.vn

³ Faculty of Information Technology, Electric Power University, Hanoi, Vietnam
hongpd@epu.edu.vn

⁴ Faculty of Automation Technology, Electric Power University, Hanoi, Vietnam
chungltk@epu.edu.vn

Abstract. Nowadays word embeddings, also known as word vectors, play an important role for many NLP tasks. In general, these word representations are learned from an unannotated corpus and they are independent from their applications. In this paper we aim to enrich the word vectors by adding more information derived from an application of them which is the aspect based sentiment analysis. We propose a new model using a combination of unsupervised and supervised techniques to capture the three kinds of information, including the general semantic distributed representation (i.e. the conventional word embeddings), and the aspect category and aspect sentiment from labeled and unlabeled data. We conduct experiments on the restaurant review data (<http://spidr-ursa.rutgers.edu/datasets/>). Experimental results show that our proposed model outperforms other methods as Word2Vec and GloVe.

1 Introduction

Word representations are a critical component of many natural language processing systems. A common way for representing words is that they are represented as indices in a dictionary, but this fails to capture the rich relational structure of the dictionary. To overcome this limitation, many studies, such as [2, 9, 13] represent each word as a real-valued vector with a low-dimensional and continuous, this known as word embeddings. These studies learn word embeddings based on contexts of words and the achieved results are words with similar grammatical usages and semantic meanings, for example, two words “excellent” and “good” are mapped into neighboring vectors in the embedding space. Since word embeddings capture semantic information, they used as inputs or a representational basis (i.e. features) for natural language processing tasks, such as text classification, document classification, information retrieval, question answering, name entity recognition, sentiment analysis, and so on.

Aspect-based sentiment analysis is a special type of sentiment analysis, it aims to identify the different aspects of entities in textual reviews or comments, and the corresponding sentiment toward them. Some tasks have been focused, such as aspect term extraction [1, 17], aspect category detection and aspect sentiment classification [1, 5, 6], aspect rating and aspect weight determination [15, 16]. Although the context-based word embeddings have been proven useful in many natural language processing tasks [4, 21], but it seems that this representation lacks of information from the task using it. Particularly, in the task of aspect-based sentiment analysis, general word embeddings can ignore the information about aspect sentiment and aspect category. For example, the given input sentence “*We stayed in a junior suite on ground level and it was spotlessly clean, great maid service and room service on tap 24/7*” labeled “Service” and “Positive” as its information of aspect category and sentiment information respectively, which will not be used in the word embedding learning algorithms as [9, 13]. The problem here is how to utilize these kinds of information and integrate it to the general word embeddings.

In this paper, we propose a new model using both labeled and unlabeled data for learning word representations. Specifically, our model contains two components, the first component learns the general word embeddings (we consider it contains semantic information in general) using an unsupervised model, here we choose the CBOW model (Mikolov et al. [9]). The second component learns word embeddings via a supervised neural network objective to the aspect category and aspect sentiment information. Different to the CBOW, in our model we first learn a general vector by using the CBOW model and then use this result as the initial value for rebuilding the word embeddings with regarding to the aspect sentiment and aspect category labels.

In the experiment, we use 52,574 reviews on the domain of restaurant products¹. We evaluate the effectiveness of word embeddings by applying them to the tasks of aspect category detection and aspect sentiment classification. Experimental results show that our proposed model outperforms others which use the well known word embeddings such as Word2Vec [9] and GloVec [13].

2 Related Work

In this section we review existing works closely related to our work. It includes word embedding techniques, word embeddings in sentiment analysis, and aspect-based sentiment analysis.

Word embeddings are first proposed in (Rumelhart et al. [19]), but the word embedding technique began development in 2003 with the neural network language model (NNLM) in (Bengio et al. [2]). It uses a neural network architecture containing four layers of language model to automatically learn word embeddings. In 2013, Mikolov et al. [9] proposed two models, namely skip-gram and continuous bag-of-words (CBOW). They improve training efficiency and learn high-quality word representations by simplifying the internal structure of the

¹ <http://spidr-ursa.rutgers.edu/datasets/>.

NNLM. The skip-gram model uses the current word to generate the context words. The CBOW model is the opposite of the skip-gram model, the current word is generated from the given context words. In 2014, Pennington et al. [13] proposed the GloVe model using a new global log-bilinear regression model for the unsupervised learning of word representations that outperforms the original models of Skip-gram and CBOW on word analogy, word similarity.

Many studies learned sentiment-specific word embeddings for sentiment analysis, they attempt to capture more information into word embeddings, such as Maas et al. [8] extended the unsupervised probabilistic model to incorporate sentiment information. Tang et al. [20] proposed three neural network models to learn word vectors from tweets containing positive and negative emoticons. Ren et al. [18] improved the models of Tang et al. [20] by incorporating topic information and learning topic-enriched multiple prototype embeddings for each word. Zhou et al. [25] proposed a semi-supervised word embedding based on the skip-gram model of Word2Vec algorithm, the word embeddings in their model can capture some information as semantic, the relations between sentiment words and aspects. In this paper, in order to learn word embeddings capturing semantic, aspect sentiment and aspect category information. We extend the unsupervised COBW model by adding a supervised component using input as labeled sentences. Each sentence is assigned two labels, i.e. aspect category and aspect sentiment.

Aspect-based sentiment analysis is the task of identifying the different aspects of entities in textual reviews or comments, and the corresponding sentiment toward them. Some tasks have been done by using lexical information and classification methods, such as aspect sentiment classifier [5, 22], aspect category detection [5, 6]. They have a limitation that they cannot capture semantic interactions between different words in sentences. Recently, some studies have been used word embeddings as input, such as Alghunaim et al. [1] using CRFsuite or SVM-HMM with word embeddings as features. Nguyen-Hoang et al. [12] used restricted boltzmann machines with word embeddings to build the joined model for both aspect term extraction and the sentiment polarity prediction task. Poria et al. [17] use word embeddings as input, they then use a deep convolutional neural network produces local features around each word in a sentence and combine these features into a global feature vector. However, most these studies have been used the learned word embeddings from Word2Vec model [9] which only captures word semantic relations and ignore supervised information such as aspect category and aspect sentiment, that is our objective in this paper.

3 The Proposed Method

In this section, we present a new model for learning word embeddings, which captures semantic, aspect sentiment and aspect category information. We named this model as Semantic-Sentiment-Category Word Embedding (SSCWE) model. The SSCWE model contains two components, namely semantic word embedding component and sentiment-category word embedding component. We consider

each component as a model and construct a cost function for each model. We then construct the cost function of our model by a sum of two cost functions of models.

3.1 Problem Definition

Given a sentence set $D = \{d_1, d_2, \dots, d_{|D|}\}$ extracting from a collection of reviews in a particular domain (e.g. restaurant). Let $V = \{\omega_1, \omega_2, \dots, \omega_{|V|}\}$ be a constructed dictionary from D . We need to identify a word embedding matrix $W \in \mathbb{R}^{n \times |V|}$, where the i -th column of W is the n -dimensional embedded vector for word i -th, w_i in V . Note that in our work, D contains both labeled sentences and unlabeled sentences.

3.2 Semantic Word Embedding Component

This component uses the unsupervised CBOW model in [10] to learn word embeddings capturing semantic of words. The idea of the CBOW model is to predict a target word based on its contexts. For a sentence $d \in D$ containing p words, $d = (\omega_1, \omega_2, \dots, \omega_p)$. Let $\omega_i \in d$ be the target word, which is predicted based on the context h ,

$$h = (\omega_{(i-C)}, \dots, \omega_{(i-1)}, \omega_{(i+1)}, \dots, \omega_{(i+C)}) \quad (1)$$

where C is a size of context. The CBOW model takes the one-hot vectors of context words in h as input and the one-hot vector of the word ω_i as output, the architecture of the CBOW model as in Fig. 1.

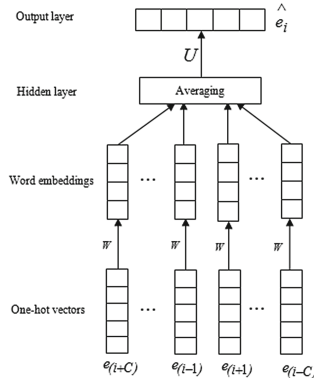


Fig. 1. An illustration of the semantic word embedding component using CBOW model

At the hidden layer, the weights are shared for all words by the average of the vectors of the input context words as follows

$$v(h) = \frac{W \cdot (e_{(i-C)} + \dots + e_{(i-1)} + e_{(i+1)} + \dots + e_{(i+C)})}{2C} \quad (2)$$

where $W \in R^{n \times |V|}$ is the weight matrix (i.e. word embedding matrix) between the input layer and the hidden layer, $e_{(i-C)}, \dots, e_{(i+C)}$ are the corresponding one-hot vectors of words in the context.

The one-hot vector of word w_i is computed according to the model as follows:

$$\hat{e}_i = g(U.v(h) + u_0) \quad (3)$$

where g is a softmax function, $U \in R^{|V| \times n}$ is a weight matrix between the hidden layer and the output layer, $u_0 \in R^{|V|}$ is a bias vector.

The cross entropy cost function for the pair of the context and the target word is defined as follows:

$$H(\hat{e}_i, e_i) = - \sum_{j=1}^{|V|} e_{ij} \log \hat{e}_{ij} \quad (4)$$

The cross entropy cost function for the data set D is defined as follows:

$$E(\theta) = \sum_{d \in D} \sum_{i=1}^{N_d} H(\hat{e}_i, e_i) \quad (5)$$

Where $\theta = \{W, U, u_0\}$ is a parameter set of the component, N_d is the total number of contexts in the sentence d .

3.3 Sentiment-Category Word Embedding Component

The semantic word embedding component using CBOW model only capture semantic information from the given data set. In this component, we extend it with a supervised aspect category and sentiment component. We take some labeled sentences and word embeddings (i.e. matrix W) as input to learn a prediction model for both aspect category and sentiment. Our goal is to help word embeddings capturing aspect category and sentiment information in each labeled sentence. We named this component as sentiment-category word embedding model, its architecture is shown in Fig. 2.

Let $D' = \{d_1, d_2, \dots, d_{|D'|}\} \subset D$ be a collection of all labeled sentences of D , each textual sentence is assigned two labels, i.e. aspect sentiment and aspect category. Let k be the number of aspect category labels and m be the number of aspect sentiment labels in D' . For a sentence $d \in D'$ containing p words, the word embedding vector of word i -th is computed as follows:

$$x_i = W.e_i \quad (6)$$

where $W \in R^{n \times |V|}$ be an embedding matrix of words in V , e_i is a one-hot vector of the word i -th.

The vector of sentence d at the hidden layer is computed by averaging word embedding vectors of the input words as follows:

$$v(d) = \frac{1}{p} \sum_{i=1}^p x_i \quad (7)$$

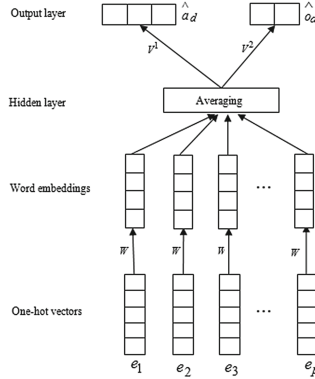


Fig. 2. An illustration of the sentiment-category word embedding model learns to capture aspect sentiment and aspect category information.

The aspect category vector of sentence d is computed as follows:

$$\hat{a}_d = g(V^1.v(d) + b^1) \quad (8)$$

The sentiment vector of sentence d is computed as follows:

$$\hat{o}_d = g(V^2.v(d) + b^2) \quad (9)$$

where $V^1 \in \mathbb{R}^{k \times n}$, $V^2 \in \mathbb{R}^{m \times n}$ are weight matrices from the hidden layer to the output layer, $b^1 \in \mathbb{R}^k$ and $b^2 \in \mathbb{R}^m$ are bias vectors, g is a softmax function.

Let $a_d \in \mathbb{R}^k$ be a binary label vector and be the desired target vector of the aspect categories in the sentence d . Each value in a_d indicates that the sentence d is discussing an aspect category or not. Let $o_d \in \mathbb{R}^m$ be a binary label vector and be the desired target vector of the aspect sentiments in the sentence d . Each value in o_d indicates that the sentence d is discussing an aspect sentiment or not. The cross entropy cost function for the sentence d is computed as follows:

$$H(\hat{a}_d, a_d, \hat{o}_d, o_d) = -\sum_{i=1}^k a_{di} \log \hat{a}_{di} - \sum_{i=1}^m o_{di} \log \hat{o}_{di} \quad (10)$$

The cross entropy cost function for the data set D' is:

$$E(\theta) = \sum_{d \in D'} H(\hat{a}_d, a_d, \hat{o}_d, o_d) \quad (11)$$

where $\theta = [W, V^1, V^2, b^1, b^2]$ is the parameter set of the component.

3.4 Our Model Learning

In order to learn our model, we combine cross entropy cost functions of the two presented components into an overall cross entropy cost function as follows:

$$E(\theta) = \sum_{d \in D} \sum_{i=1}^{N_d} H(\hat{e}_i, e_i) + \sum_{d \in D'} H(\hat{a}_d, a_d, \hat{o}_d, o_d) + \frac{1}{2} \lambda_\theta \|\theta\|^2 \quad (12)$$

where $\theta = [W, U, u_0, V^1, V^2, b^1, b^2]$, λ_θ is the regularization parameter and $\|\theta\|^2 = \sum_i \theta_i^2$ is a norm regularization term. In order to compute the parameters θ , we apply back-propagation algorithm with stochastic gradient descent to minimize this cost function.

4 Experiments

4.1 Experimental Data and Implementation of the SSCWE Model

We use two data sets on the domain of restaurant products. The first data set contains 3,111,239 unlabeled sentences extracted from 229,907 reviews². The second data set contains 52,574 reviews (See Footnote 1), which is used in previous works [3, 5, 12, 23]. It contains six aspect category labels as *Food*, *Price*, *Service*, *Ambience*, *Anecdotes*, and *Miscellaneous*. And four aspect sentiment labels as *Positive*, *Negative*, *Neutral* and *Conflict*. Each sentence is labeled for both aspect category and sentiment.

We implemented a pre-processing tasks on the second data set that includes: removing stop words, removing low frequent words and removing the sentences with less than two words. After pre-processing the data, we get about 190,655 sentences in total. We randomly get 75% of these sentences to learn word embeddings, the remaining 25% of these sentences is used to evaluate the quality of the SSCWE model. Some statistics of the second data set are shown in Table 1.

To implement the SSCWE model, we develop an iterative algorithm based on the back-propagation algorithm to minimize the cost function in Eq. (12), and perform it with the parameters as follows: the size of context C is 4, the regularizations $\lambda_W = \lambda_U = 10^{-4}$, $\lambda_{V^1} = \lambda_{b^1} = 10^{-3}$ and $\lambda_{V^2} = \lambda_{b^2} = 10^{-4}$, the weight matrices W, U, V^1, V^2 are randomly initialized in the range of $[-1, 1]$, the bias vectors u_0, b^1, b^2 are initialized to be zero, the learning rate η is 0.025 and the iterative threshold $I = 50$.

4.2 Evaluation

We compare our SSCWE model with the following baseline models:

- **Word2Vec:** Mikolov et al. [9] develop Word2Vec with two model architectures, CBOW and skip-gram to learn word embeddings. We use here the CBOW model with the tool at <https://github.com/piskvorky/gensim/>, this model is also used for our semantic word embedding component.

² <https://www.yelp.com/datasetchallenge/>.

Table 1. Statistics of the second data set

Aspect	Num. sentences used for	
	Learning word embeddings	Evaluating word embeddings
Food	4,386	1,462
Price	44,912	14,970
Service	22,470	7,489
Ambience	17,729	5,909
Anecdotes	18,396	6,132
Miscellaneous	35,100	11,700
Total	142,993	47,662

- **GloVe** (Pennington et al. [13]): using a global log-bilinear regression model to learn word embeddings. Here we use the tool at <https://nlp.stanford.edu/projects/glove/>.
- **SCWE**: We consider the sentiment-category word embedding (SCWE) component as a variant model of the SSCWE model. It leaves out the syntactic contexts of words, uses the randomly initialized word embeddings and then modified during training.
- **SSCWE***: This model is a variant of the SSCWE model, it uses only labeled sentences to learning word embeddings.

Two models Word2Vec and GloVe learn word embeddings using the same data set and the same size of context as the SSCWE model, but they do not use aspect sentiment and aspect category information. Both the SCWE and SSCWE* models use only 142,993 labeled sentences for learning word embeddings. The size of dimensional word embeddings in all models is 300. In addition, we also use pre-trained word vectors to compare with the models. For the pre-trained word vectors learned from the Word2Vec model, we use the publicly available Word2Vec vectors³ which were trained on 100 billion words from Google News. We denote them as Pre-Word2Vec. For the pre-trained word vectors learned from the GloVe model, we use the word embedding vectors⁴ (Pennington et al. [13]), which is trained from a web based data and the vocabulary size is about 1.9 M, we denote them as Pre-GloVe.

In fact, we can not evaluate the word embeddings directly because we cannot obtain the ground-truth word embeddings from the given dataset. As a result, we choose to evaluate the word embeddings indirectly through using them as the input of a prediction model in the two tasks of aspect-based sentiment analysis, i.e. aspect category detection and aspect sentiment classification. Specifically, we use the CNN model (Kim [7])⁵ as the prediction model for these tasks. The lower prediction results in any experiment mean that the word embeddings used

³ <https://code.google.com/archive/p/Word2Vec/>.

⁴ <http://nlp.stanford.edu/projects/glove/>.

⁵ https://github.com/yoonkim/CNN_sentence.

in that case are low. In each experiment case, the CNN model performs 4-fold cross validation with 47,662 sentences. We then get the average value of metrics to report. The metrics are used to measure the prediction results as F1 score⁶ and Accuracy.

Aspect Category Detection. In Table 2, we show the achieved F1-score of aspect category detection task for each method. In a general observation, our SSCWE model performs better than other models. This indicates that the learned word embeddings from the SSCWE model is better than other models for the specific tasks. For the semantic word embeddings, we show that the trained semantic word embeddings (i.e. Word2Vec and GloVe) on the restaurant review data is slightly better than the pre-trained word embeddings (i.e. Pre-Word2Vec and Pre-GloVe). For the sentiment-category word embeddings, although the SCWE (Sentiment-Category Word Embedding) model is a version of the SSCWE model, it only captures aspect sentiment and aspect category information, but it outperforms Word2Vec and GloVe model. This indicates that the aspect sentiment and aspect category information play an important role in learning word embeddings.

Table 2. Results of aspect category detection with different embedding word learning methods

Method	Precision	Recall	F1 score
Word2Vec	79.21	77.88	78.54
GloVec	80.53	77.89	79.19
Pre-Word2Vec	78.13	76.37	77.24
Pre-GloVec	79.67	78.36	79.01
Our SCWE	79.93	80.15	80.04
Our SSCWE*	82.56	81.68	82.12
Our SSCWE	83.16	82.38	82.77

Aspect Sentiment Prediction. In Table 3, we show the achieved F1-score of each aspect sentiment label and the accuracy metric on aspect sentiment classifier task for each method. In most methods, our two models, i.e. SCWE and SSCWE, perform better than other models. This indicates that the learned word embeddings from our models can help improving the aspect sentiment predicted results.

Querying Words. We also evaluate the quality of the word embeddings through query words. Given some sentiment words or aspect words, we can find out the most similar words with them. In Table 4, we show the interesting results of the most similar words to four given words “good”, “bad”, “food”, and “price”. Two models GloVe and SSCWE capture broad semantic similarities. The SCWE model capture the true semantic of two aspect words “food” and “price”, but

⁶ <https://en.wikipedia.org/wiki/Precisionandrecall>.

Table 3. Results for the aspect sentiment prediction task

Method	Positive-F1	Negative-F1	Neutral-F1	Conflict-F1	Accuracy
Word2Vec	86.93	52.25	66.60	55.93	79.22
GloVec	87.10	51.07	71.02	57.85	80.35
Pre-Word2Vec	82.04	49.53	68.44	53.16	79.01
Pre-GloVec	83.95	53.04	65.04	54.04	80.13
Our SCWE	89.54	64.00	74.01	56.30	81.41
Our SSCWE*	93.78	63.81	76.58	61.93	83.85
Our SSCWE	93.80	64.70	76.13	63.02	84.69

Table 4. Each target word is given with its five most similar words using cosine similarity of the vectors determined by each model

	Good	Bad	Food	Price
GloVe model	Excellant	Poor	Props	Prices
	Decent	Awful	Postings	Pricing
	Fantastic	Horrible	Vary	Penny
	Costco	Terrible	Gosh	Albeit
	Parm	Lousy	Cuisine	Praise
Our SCWE model	Delight	Horrible	Snacks	Prices
	Goodness	Alien	Restaurant	Pricing
	Millionaires	Calling	Hospitality	Pricey
	Paycheck	Deveined	Fashion	150
	Best	Discriminating	Soooooo	500
Our SSCWE model	Excellent	Terrible	Meal	Prices
	Great	Worse	Foods	Pricey
	Better	Poor	Snacks	Pricing
	Wonderfull	Lousy	Variation	Bills
	Unbeatable	Aweful	Cuisine	Buck

it does not capture the good meaning of the two sentiment words “good” and “bad”, this is because the SCWE model does not use the semantic of input text. The full SSCWE model captures the three information kinds, i.e. semantic, aspect category and aspect sentiment, thus it seems to be better than GloVe and SCWE. This comparison again indicates that adding aspect category and sentiment information in word embeddings is importance.

Effect of the dimensional word embeddings. In order to evaluate in detail the influence of the dimensional word vector. We chose the size of dimensional word vector from 50 to 500. In Fig. 3 shows the achieved metric, we see that our models slightly better than other models in all cases. The models Word2Vec and

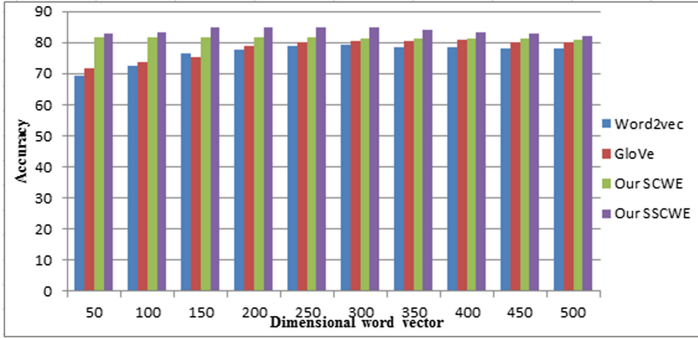


Fig. 3. Effect of the dimensional word embedding on aspect sentiment prediction

GloVe perform well when the dimensional word embedding is above 200. Our models SCWE and SSCWE perform best with the dimensional word embedding is 150, they do not improve the prediction results when increasing the dimensional word embedding from 200 to 500.

5 Conclusion

In this paper, we have proposed a new model using a combination of unsupervised and supervised techniques to learn word embeddings for the task of aspect based sentiment analysis. Our model has utilized the advantages of the general Word2Vec model (i.e. CBOW) integrated with the supervised information from specific tasks. As the result the model has the ability to capture the three kinds of information including semantic, aspect category and aspect sentiment. Various experiments have been conducted and confirmed the effectiveness of the proposed model. Experimental results shows that the tasks of aspect sentiment and category classifications have been improve, as well as our proposed model outperforms the Word2Vec and GloVe models.

Acknowledgement. This paper is supported by The Vietnam National Foundation for Science and Technology Development (NAFOSTED) under grant number 102.01-2014.22.

References

1. Alhunaim, A., Mohtarami, M., Cyphers, S., Glass, J.: A vector space approach for aspect based sentiment analysis. In: Proceedings of NAACL-HLT 2015, pp. 116–122 (2015)
2. Bengio, Y., Ducharme, R., Vincent, P., Janvin, C.: A neural probabilistic language model. *J. Mach. Learn. Res.* **3**, 1137–1155 (2003)
3. Brody, S., Elhadad, N.: An unsupervised aspect-sentiment model for online reviews. In: Proceedings of NAACL-HLT, pp. 804–812 (2010)

4. Collobert, R., Weston, J.: A unified architecture for natural language processing. In: Proceedings of the ICML, pp. 160–167 (2008)
5. Ganu, G., Elhadad, N., Marian, A.: Beyond the stars: improving rating predictions using review text content. In: Proceedings of WebDB, pp. 1–6 (2009)
6. Kiritchenko, S., Zhu, X., Cherry, C., Mohammad, S.M.: NRC-Canada-2014: detecting aspects and sentiment in customer reviews. In: Proceedings of SemEval, pp. 437–442 (2014)
7. Kim, Y.: Convolutional neural networks for sentence classification. In: Proceedings of EMNLP, pp. 1746–1751 (2014)
8. Maas, A.L., Daly, R.E., Pham, P.T., Huang, D., Nguyen, A.Y., Potts, C.: Learning word vectors for sentiment analysis. In: Proceedings of ACL, pp. 142–150 (2011)
9. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Proceedings of NIPS, pp. 1–9 (2014)
10. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space, CoRR (2013)
11. Mnih, A., Hinton, G.E.: A scalable hierarchical distributed language model. In: Proceedings of NIPS, pp. 1081–1088 (2008)
12. Nguyen-Hoang, B.D., Ha, Q.V., Nghiem, M.Q.: Aspect-based sentiment analysis using word embedding restricted Boltzmann machines. In: Proceedings of CSoNet 2016, pp. 285–297 (2016)
13. Pennington, J., Socher, R., Manning, C.D.: GloVe: global vectors for word representation. In: Proceedings of EMNLP, pp. 1532–1543 (2014)
14. Pavlopoulos, J., Androutsopoulos, I.: Aspect term extraction for sentiment analysis: new datasets, new evaluation measures and an improved unsupervised method. In: Proceedings of ACL, pp. 44–52 (2014)
15. Pham, D.H., Le, A.C., Le, T.K.C.: A least square based model for rating aspects and identifying important aspects on review text data. In: Proceedings of NICS, pp. 16–18 (2015)
16. Pham, D.H., Le, A.C., Nguyen, T.T.T.: Determining aspect ratings and aspect weights from textual reviews by using neural network with paragraph vector model. In: Proceedings of CSoNet, pp. 309–320 (2016)
17. Poria, S., Cambria, E., Gelbukh, A.: Aspect extraction for opinion mining with a deep convolutional neural network. *Knowl. Based Syst.* **108**, 42–49 (2016)
18. Ren, Y., Zhang, Y., Zhang, M., Ji, D.: Improving Twitter sentiment classification using topic-enriched multi-prototype word embeddings. In: Proceedings of AAAI, pp. 3038–3044 (2016)
19. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagating errors. *Nature* **323**, 9 (1986)
20. Tang, D., Qin, B., Liu, T.: Learning sentiment-specific word embedding for Twitter sentiment classification. In: Proceedings of ACL, pp. 1555–1565 (2014)
21. Turian, J., Ratinov, L., Bengio, Y.: Word representations: a simple and general method for semisupervised learning. In: Proceedings of the ACL, pp. 384–394 (2010)
22. Wagner, J., Arora, P., Cortes, S., Barman, U., Bogdanova, D., Foster, J., Tounsi, L.: DCU: aspect based polarity classification for semeval task 4. In: Proceedings of SemEval, pp. 223–229 (2014)

23. Wang, L., Liu, K., Cao, Z., Zhao, J., Melo, G.D.: Sentiment-aspect extraction based on restricted Boltzmann machines. In: Proceedings of ACL, pp. 616–625 (2015)
24. Zhao, W.X., Jiang, J., Yan, H., Li, X.: Jointly modeling aspects and opinions with a MaxEnt-LDA hybrid. In: Proceedings of EMNLP, pp. 56–65 (2010)
25. Zhou, X., Wan, X., Xiao, J.: Representation learning for aspect category detection in online reviews. In: Proceedings of AAAI, pp. 417–423 (2015)



BolLy: Annotation of Sentiment Polarity in Bollywood Lyrics Dataset

G. Drushti Apoorva and Radhika Mamidi(✉)

NLP-MT Lab, KCIS, IIIT-Hyderabad, Hyderabad, India
drushti.g@research.iiit.ac.in, radhika@iiit.ac.in

Abstract. This work presents a corpus of Bollywood song lyrics and its metadata, annotated with sentiment polarity. We call this *BolLy*. It contains lyrics of 1055 songs ranging from those composed in the year 1970 to the most recent ones. This dataset is of utmost value as all the annotation is done manually by three annotators and this makes it a very rich dataset for training purposes. In this work, we describe the creation and annotation process, content, and the possible uses of the dataset. As an experiment, we have built a basic classification system to identify the emotion polarity of the song based solely on the lyrics and this can be used as a baseline algorithm for the same. *BolLy* can also be used for studying code-mixing with respect to lyrics.

1 Introduction

Bollywood Hindi songs constitute a major part of music sales [25] in India and these are widely available in digitized form on the Internet. People choose to listen to different types of songs depending on their mood. This highlights the fact that classification of music on the basis of emotions they evoke [23] would help in varied tasks. Research in this field would most definitely require annotated datasets. Our work provides this resource.

In the dataset we present, the songs have been classified into two classes but for the definition of these classes, we have taken into consideration a wide variety of emotions. The size of *BolLy* is also substantially large as compared to other such datasets existing for Bollywood songs with their primary language being Hindi. It helps address the differences in Hindi and Western music appropriately [17]. A unique feature of this resource is that it is in the Devanagari script. This avoids the preprocessing cost of text normalisation. It contains the lyrics and metadata of 1055 Bollywood songs. This would be very useful for research in fields related to lyrics analysis and emotion polarity detection.

The dataset presented in this work has varied applications. One of the major applications is to extract emotion polarity from song lyrics which can be used in the creation of various systems such as automatic playlist generation and recommendation systems. They would also aid in music library management. ‘BolLy’ dataset contains Hindi as the major language and also has the usage of

English or other Indian languages which is an emerging trend in the songs being composed. Hence, another application can be in the field of code mixing.

Our work is a step towards providing quality dataset for research in fields mentioned above. The purpose of this dataset is:

- to encourage research in the field of emotion extraction, particularly in lyrics.
- to provide a reference dataset for evaluating research.
- to help new researchers get started in the field of Hindi lyrics analysis.
- to provide a kickstart to studies on code mixing in Bollywood songs.

This paper is organised as follows: Sect. 2 elaborates the related work in this domain while Sect. 3 talks about the dataset, its creation and annotation. As mentioned earlier, there can be various applications of such a dataset. An experiment of the usage of this dataset for song classification based on emotion polarity, which can be used as a baseline for such studies, is described in Sect. 4. Section 5 discusses the further work possible on this dataset and has the conclusion of the work presented.

2 Related Work

One of the major contributions of the dataset proposed is in the field of music classification. Some of the important work done towards classification of music is mentioned. This is followed by discussion of related work specific to available datasets in this context or the taxonomies used for annotation.

The work done in music classification involves those making use of lyrics [6], audio [3, 12, 13, 15] or a multimodal approach [2, 9, 10]. The classification techniques based on lyrics have been investigated in various languages. Affective lexicon and fuzzy clustering method were used for Chinese song lyrics in the work presented in [6]. In [5], it was shown that audio features do not always outperform lyric features using a dataset of 8,784 English song lyrics for the given task.

2.1 Lyrics Datasets

Since abundant resources are available for English song lyrics, we would restrict our discussion to datasets created specifically for Hindi and Indian languages. The dataset available for Bollywood song lyrics in [16] has a total of 461 song lyrics classified as positive or negative based on certain moods in each class.

The work presented in [20] uses another dataset for Bollywood song lyrics that contains lyrics and the metadata of 300 songs composed between the years 1995 and 2015 with details about the usage of foreign words in different decades. A Telugu dataset of 300 song lyrics has been used in the work towards a multimodal approach for sentiment analysis presented in [1]. Thus, to the best of our knowledge, the dataset presented by us, *BolLy*, is the largest of its kind amongst other resources for the language and the only one that contains instances of code-mixing.

2.2 Taxonomy for Classification

A proper taxonomy is vital to mood classification of songs. The Russell’s Circumplex Model of 28 Affect Words [19] is one such two dimensional model that classifies moods using dimensions denoted as ‘pleasant-unpleasant’ and ‘arousal-sleep’. It is seen that a lot of research groups have adapted Russell’s model or used subsets of it for their work [7, 22, 26]. Our work makes use of this model to decide the scope of the tags used for annotation. This is discussed in detail in Sect. 3.2 (2) that deals with annotation guidelines.

3 The Dataset

3.1 Creation of the Dataset

Bollywood lyrics can be easily extracted from many websites. They are commonly available transliterated in the Roman script. For simplicity of processing, we extracted them from a source where Hindi words were available in Devanagari script, i.e. consisting of utf-8 characters. The lyrics of the songs were extracted with metadata including the movie or album they belong to, the singers and the year of release.

After the initial collection of data, it was cleaned to further reduce pre-processing required. Repetitions of lines or words in the lyrics were represented using numbers, for example, a line was followed by ‘X2’ if it was to be repeated twice. In certain cases, these numbers were in Devanagari. All such representations were removed and the line or word in question were copied as many times mentioned.

In Bollywood, a lot of songs are remixed into different versions. Sometimes the same songs are sung by different singers. There are quite a few instances wherein the same song occurs in different moods, such as both happy and sad, in a movie. All such songs appeared multiple times in the dataset. These songs differ in terms of audio features and evoke varied emotions when listened to. As there was no difference in their lyrics and our work focuses solely on the emotions evoked by song lyrics, the multiple occurrences of such songs were removed from the dataset, and only one file was retained.

Following are the features of the dataset:

- originally procured 1,082 song lyrics.
- 1,055 song lyrics in the final dataset, after removal of duplicates.
- song lyrics/files with metadata amounting to 2.6 MB data.
- 712 songs annotated as positive in the final dataset.
- 343 songs annotated as negative in the final dataset.
- total number of tokens in the dataset are 2,17,285.
- average number of tokens in a song are rounded off to 211.
- total number of tokens in positive songs are 1,51,362.
- average number of tokens in a positive song can be rounded off to 218.
- total number of tokens in negative songs are 65,923.
- average number of tokens in a negative song can be rounded off to 196.

3.2 Annotation

Principles of Annotation. Three levels of granularity are described for existing methods of sentiment analysis in [11]. On the basis of the level defined, the task is to identify if positive or negative sentiment is expressed at that level. They can be carried out at the level of the whole document as in [24], or at sentence level or at the level of entities and aspects [4, 21]. It is possible that different smaller parts of a song’s lyrics evoke different emotions. We aim to identify whether the whole song’s lyrics evoke a positive or a negative emotion. Hence, it is best for us to look at document level classification. The annotators were asked to go through the whole song before tagging them. This results in the tag corresponding to the polarity of the general mood evoked by it.

Annotation Process. Each song in the dataset was annotated as positive or negative by three annotators, all of whom were university students in the age group 20–24 and were native speakers of Hindi. Songs evoke a certain emotion or mood, and these can be classified as those with positive or negative valence according to Russell’s Circumplex Model, as shown in Fig. 1. The songs that evoke emotions ranging from ‘aroused’ to ‘sleepy’ including ‘calm’, ‘satisfied’, ‘delighted’, ‘excited’, etc. are to be tagged as positive. Negative tags are to be given to songs evoking moods such as ‘angry’, ‘annoyed’, ‘miserable’, ‘depressed’, etc., all of them spanning from ‘alarmed’ to ‘tired’. Each song is annotated with the tag given by majority of the annotators for it.

The annotations were carried out in a controlled environment in which the annotators were not allowed to listen to the audio of the song presented. Hence, the annotation is solely on the basis of lyrics. Also, the songs were presented to the annotators without any of the metadata associated with it to prevent any preconditioning. The number of positive and negative tags given by each annotator can be seen in Table 1.

Table 1. Number of positive and negative tags given by each annotator.

Annotator	Positive tags	Negative tags
1	721	334
2	728	327
3	710	345

By the end of annotation of the final dataset of 1055 songs, 712 are tagged as positive while the rest 343 are annotated as negative. From the original dataset of 1,082 songs, 27 were removed as they were duplicates. The annotation for these 27 songs were used to check for consistency of the individual annotators. While annotators 1 and 2 had annotated the duplicate instances of only 1 song out of 27 differently, the third annotator’s tagging was inconsistent for 2 songs. These small numbers can be ignored in such a large dataset as they show that the annotators were rarely inconsistent in their task.

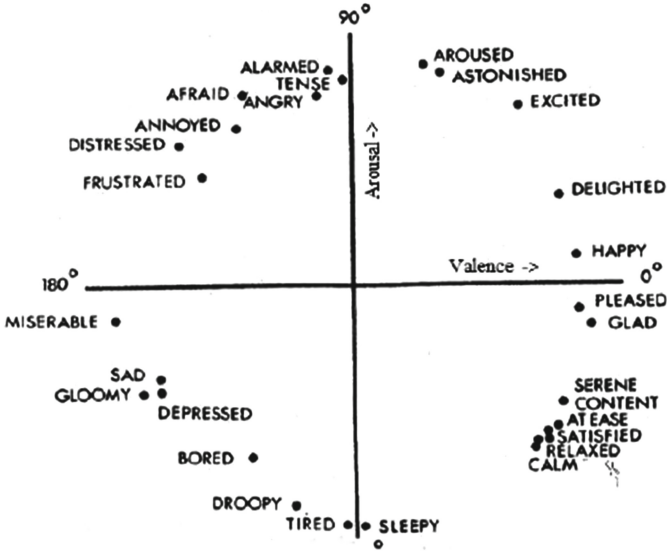


Fig. 1. Russell's Circumplex Model [19] classifying 28 affect words on the basis of positive and negative valence and arousal.

Inter-annotator Agreement. Inter-annotator agreement is a measure of how well the annotators can make the same annotation decision for the same category. Given the task in hand, it is fair to assume that annotation of the songs based on the emotions evoked by reading the lyrics is a very subjective opinion. Thus, inter-annotator agreement becomes an important factor in validating the annotators' work.

There are many statistical measures that can be used for measuring the reliability of agreement between the annotators given the number of data points, number of annotators and the number of tags they can be given. Cohen's kappa, Fleiss' kappa and Scott's pi are some of them. Out of these, Cohen's kappa and Scott's pi work only for cases where there are two annotators. Fleiss' kappa, a generalisation of Scott's pi statistic, is useful when there are more annotators and thus is most suited for our work.

The Fleiss' kappa obtained for the annotations for our dataset is 0.79. This corresponds to 'substantial agreement' [8] according to the interpretation of Fleiss' kappa shown in Table 2.

4 Experimentation

4.1 Theory

We conducted a few experiments to show one of the applications of the dataset. This involved classification of the song lyrics to extract sentiment polarity expressed by them. This can be used as a baseline experiment for sentiment

Table 2. Interpretation of Fleiss’ kappa values for inter-annotator agreement [8].

κ	Interpretation
<0	Poor agreement
0.01–0.20	Slight agreement
0.21–0.40	Fair agreement
0.41–0.60	Moderate agreement
0.61–0.80	Substantial agreement
0.81–1.00	Almost perfect agreement

analysis tasks for Hindi song lyrics. We make use of Naive Bayes [14] and Support Vector Machine classifiers for these experiments. These are discussed here briefly.

Naive Bayes methods are supervised learning algorithms based on the Bayes’ Theorem with the assumption that every pairs of features are independent. This leads to using the classification rule $P(y|x_1\dots x_n) \propto P(y) \prod_{i=1}^n P(x_i|y)$, given a class variable y and dependent feature vector x_1 through x_n . There are different variations of this depending on the distribution of the data points.

Another supervised learning algorithm is SVM or Support Vector Machine. This makes use of a hyperplane to identify the decision boundary. Given a dataset of n points of the form $(x_1, y_1)\dots(x_n, y_n)$ where y_i is either 1 or -1 depending on which class the data point x_i belongs to. The hyperplane can be defined as the set of points x satisfying $w \cdot x - b = 0$ where w is the normal vector to the hyperplane and parameter $\frac{b}{\|w\|}$ expresses the offset of this hyperplane along w from the origin.

4.2 Methodology

The song lyrics were available as files which were converted into separate lists of lyrics and their corresponding tags for positive and negative songs, called *PLyr* and *PTags* for positive songs and *NLyr* and *NTags* for negative songs, respectively. For splitting the dataset into training and testing set, we create lists *TrainLyr* and *TestLyr* with their respective tags in the lists *TrainTags* and *TestTags*. This split is in the ratio 9:1. *TrainLyr* and *TrainTags* are used to train the models for Multinomial Naive Bayes (MultinomialNB), Bernoulli’s Naive Bayes (BernoulliNB) and Support Vector Machine (SVM). The latter is run over 5 folds. These trained models are used to predict the sentiment polarity of the data points in *TestLyr* and the predicted tags are stored in *PredTags*, which are then compared with *TestTags* for evaluation.

4.3 Experiments Conducted

The method explained above was conducted on the *BolLyr* dataset. The dataset was split in the ratio of 9:1 for training and testing purposes. As the class is

imbalanced and there are more number of positively tagged songs, we set the class prior according to the class distribution. Without this, the classifier would have tagged all the test data points as positive as that would give a better result but that is not our aim. Specifying the class priors helps the classifier to take into account the features given and predict the classes for test data accordingly.

These experiments were carried out using an open source Python library, ‘scikit-learn’ [18]. The features used included bag of words, term frequency and term frequency-inverse document frequency which were all extracted using the modules from this library. The evaluation metrics such as accuracy, precision, recall and F1 measure were extracted. The SVM classifier was run for 10 folds over the dataset.

4.4 Results and Discussion

The average accuracies for the three classifiers with the data split in the ratio 9:1 are shown in the Table 3. Subsequent tables show the other evaluation metrics for each of the classifiers used (Table 4).

Table 3. Accuracies obtained for the classifiers

Classifier used	Accuracy (%)
MultinomialNB	69.61
BernoulliNB	71.57
SVM	75.49

Table 4. MultinomialNB classifier scores

Classifier	Class	Precision	Recall	F1 score (%)
MultinomialNB	Positive	0.80	0.62	0.70
	Negative	0.46	0.67	0.54
BernoulliNB	Positive	0.80	0.77	0.79
	Negative	0.56	0.61	0.58
SVM	Positive	0.74	0.99	0.84
	Negative	0.90	0.27	0.42

These results show that Support Vector Machine works the best for this experiment. These experiments can be used as a baseline as the features used are very basic and not tuned specific to the dataset or the classification problem. Incorporating these would definitely give better results which is why the results shown here would work well as a baseline. Precision can be looked at as a classifier’s exactness while recall would give a measure of its completeness. If we look at all the evaluation metrics, we see that barring two cases, the models give better precision and recall for the positive class.

5 Future Work and Conclusion

Detection and classification of sentiments and opinion mining is a challenging task from the point of view of computational linguistics. This is because opinions are not realised uniformly in the syntax, semantics and the pragmatics of language. Even more difficult is this task in song lyrics as there is no fixed grammar or structure that is followed. Our work is a contribution towards encouraging work in this direction.

As mentioned, a variety of applications are possible once the sentiment or mood of songs are identified. Some of them include automatic playlist generation, digital music library management, song suggestion systems, etc. The dataset presented, *BolLy* would indeed thrust forward the studies conducted in the related fields and prove to be better than existing resources in Hindi.

The experiment shown is just a basic implementation of an opinion extraction task and gives a good baseline for specific and fine-tuned implementations for the same. It gives an insight as to how imbalanced datasets can be handled and what classification algorithms would serve our purpose better.

The insights derived would be helpful to build a better and more accurate system for opinion extraction from songs, using existing resources like SentiWordNet, subjectivity lexicon, language identifiers to identify and extract better features. Another possible work would be to assess the dynamic change of mood over the period of song and compare it with the general emotion polarity of the whole song.

Further, this dataset can be annotated and used for code-mixing tasks as well. It can also be used for cueing in song generation systems. This dataset once used for training a specific model, would be critical in dynamic collection of song lyrics and their annotation, thus making available a substantially large, rich corpus for similar tasks. This would help creating an invaluable resource.

References

1. Abburi, H., Akkireddy, E.S.A., Gangashetty, S.V., Mamidi, R.: Multimodal sentiment analysis of Telugu songs. In: Proceedings of the 4th Workshop on Sentiment Analysis Where AI Meets Psychology (SAAIP 2016), pp. 48–52 (2016)
2. Bischoff, K., Firan, C.S., Paiu, R., Nejd, W., Laurier, C., Sordo, M.: Music mood and theme classification—a hybrid approach. In: ISMIR (2009)
3. Fu, Z., Lu, G., Ting, K.M., Zhang, D.: A survey of audio-based music classification and annotation. *IEEE Trans. Multimedia* **13**, 303–319 (2011)
4. Hu, M., Liu, B.: Mining and summarizing customer reviews. In: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 168–177. ACM (2004)
5. Hu, X., Stephen Downie, J., Laurier, C., Bay, M., Ehmann, A.F.: The 2007 MIREX audio mood classification task: lessons learned. In: Proceedings of the 9th International Conference on Music Information Retrieval (2008)
6. Hu, Y., Chen, X., Yang, D.: Lyric-based song emotion detection with affective lexicon and fuzzy clustering method. In: ISMIR (2009)

7. Katayose, H., Imai, M., Inokuchi, M.: Sentiment extraction in music. In: 9th International Conference on Pattern Recognition (1988)
8. Landis, J.R., Koch, G.G.: The measurement of observer agreement for categorical data. *Biometrics* **33**, 159–174 (1977)
9. Laurier, C., Grivolla, J., Herrera, P.: Multimodal music mood classification using audio and lyrics. In: Seventh International Conference on Machine Learning and Applications, ICMLA 2008, pp. 688–693. IEEE (2008)
10. Laurier, C., Sordo, M., Herrera, P.: Mood cloud 2.0: music mood browsing based on social networks. In: Proceedings of the 10th International Society for Music Information Conference (ISMIR 2009), Kobe, Japan (2009)
11. Liu, B.: Sentiment analysis and opinion mining. *Synth. Lect. Hum. Lang. Technol.* **5**(1), 1–167 (2012)
12. Lu, L., Liu, D., Zhang, H.-J.: Automatic mood detection and tracking of music audio signals. *Trans. Audio Speech Lang. Process.* **14**, 5–18 (2006)
13. Lie, L., Liu, D., Zhang, H.-J.: Automatic mood detection and tracking of music audio signals. *IEEE Trans. Audio Speech Lang. Process.* **14**(1), 5–18 (2006)
14. McCallum, A., Nigam, K., et al.: A comparison of event models for Naive Bayes text classification. In: AAAI-98 Workshop on Learning for Text Categorization, vol. 752, pp. 41–48. Citeseer (1998)
15. Patra, B.G., Das, D., Bandyopadhyay, S.: Automatic music mood classification of Hindi songs. In: Proceedings of the 3rd Workshop on Sentiment Analysis Where AI Meets Psychology (2013)
16. Patra, B.G., Das, D., Bandyopadhyay, S.: Mood classification of Hindi songs based on lyrics. In: Proceedings of the 12th International Conference on Natural Language Processing (ICON 2015) (2015)
17. Patra, B.G., Das, D., Bandyopadhyay, S.: Multimodal mood classification - a case study of differences in Hindi and western songs. In: COLING (2016)
18. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al.: Scikit-learn: machine learning in python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2011)
19. Russell, J.A.: A circumplex model of affect. *J. Pers. Soc. Psychol.* **39**, 1161–1178 (1980)
20. Shakoor, A.A., Sahebodin, W.B., Pudaruth, S.: Exploring the evolutionary change in bollywood lyrics over the last two decades. In: The Second International Conference on Data Mining, Internet Computing, and Big Data (BigData 2015), p. 46 (2015)
21. Szabó, M.K., Vincze, V., Simkó, K.I., Varga, V., Hangya, V.: A Hungarian sentiment corpus manually annotated at aspect level (2016)
22. Thayer, R.E.: *The Biopsychology of Mood and Arousal*. Oxford University Press, New York (1989)
23. Trohidis, K., Tsoumakas, G., Kalliris, G., Vlahavas, I.P.: Multi-label classification of music into emotions. In: ISMIR, vol. 8, pp. 325–330 (2008)
24. Turney, P.D.: Thumbs up or thumbs down?: Semantic orientation applied to unsupervised classification of reviews. In: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, pp. 417–424. Association for Computational Linguistics (2002)

25. Ujlambkar, A.M., Attar, V.Z.: Automatic mood classification model for Indian popular music. In: 2012 Sixth Asia Modelling Symposium (AMS), pp. 7–12. IEEE (2012)
26. Yang, Y.-H., Lin, Y.-C., Cheng, H.-T., Liao, I.-B., Ho, Y.-C., Chen, H.H.: Toward multi-modal music emotion classification. In: Huang, Y.-M.R., Xu, C., Cheng, K.-S., Yang, J.-F.K., Swamy, M.N.S., Li, S., Ding, J.-W. (eds.) PCM 2008. LNCS, vol. 5353, pp. 70–79. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-89796-5_8



Frame-Based Semantic Patterns for Relation Extraction

Angrosh Mandya^(✉), Danushka Bollegala, Frans Coenen, and Katie Atkinson

Department of Computer Science, University of Liverpool, Liverpool, UK
{angrosh,danushka,coenen,katie}@liverpool.ac.uk

Abstract. This paper presents novel frame-based semantic patterns, exploiting *frame element* and *frame* annotations, provided by FrameNet for relation extraction. The proposed frame-based patterns are evaluated against state-of-the-art dependency based syntactic patterns and lexico-syntactic patterns, on three independent datasets that differ in size and construction. The results show that the proposed frame-based patterns significantly improve performance, both in terms of scoring higher precision and higher recall for relation extraction, in comparison to dependency and lexico-syntactic patterns on all three datasets.

1 Introduction

Pattern-based information extraction methods have a long and established history as a successful approach for domain-specific relation extraction [9, 15]. Although lexico-syntactic patterns are useful for relation extraction [9], these patterns suffer from two distinct problems: (i) *recall problem*: where only a small subset of trained patterns are applied on newer sentences, failing to extract information from a large number of sentence; and (ii) *precision problem*: where significant proportion of extracted information is unreliable, due to extracting incorrect entities for related relations. For example, a lexical pattern such as “*announced the products of*” learnt during the training phase from the sentence “The CEO of the company, **Steve Jobs** announced the products of **Apple** at WWDC”, when applied on a test sentence such as “Today, **Amazon** announced the products of **Apple** on their website”, results in extracting wrong entities (*Amazon, Apple*) for the CEO-COMPANY relation.

Generalizing patterns using dependency based syntactic parse are shown to be useful in improving recall of the applied pattern set [11]. Patterns using the shortest path between entities in the dependency tree [19] removes significant lexical information to generalize patterns that achieve higher recall. However, dependency patterns are observed to achieve lower precision against lexical patterns [19].

Besides dependency patterns, FrameNet [7] based frame annotations for words in sentences are useful for defining patterns. FrameNet provide annotations in terms of *semantic frames* comprising *frame elements* (FE) and *lexical units* (LUs) [7]. Further, SEMAFOR [6], FrameNet based semantic parser helps

in automatically obtaining frame annotations for newer sentences. For example, annotations provided by SEMAFOR¹ for an example sentence is as shown below:

[Lee]_{SELLER} sold [a textbook]_{GOODS} [to Abby]_{BUYER}.

In the example sentence above, *frame element* annotations such as SELLER, GOODS, BUYER are provided for different *lexical units* in the sentence. These frame elements are associated with a broader *semantic frame* COMMERCIAL_SELL, which is also provided by SEMAFOR. Using this mapping between *semantic frames* and *frame elements* pattern such as “COMMERCIAL_SELL SELLER GOODS BUYER” can be used to extract related entities (*Lee, Abby*) for the relation SELLER-BUYER. Such patterns have greater advantage over lexical and dependency-based patterns, by providing (a) higher precision through extracting specific related entities; and (b) provide better generalisation through removing lexical information. In other words, *these patterns help in achieving higher precision without losing on recall*. Following this motivation, we propose frame-based semantic patterns for relation extraction, exploiting FrameNet annotations. We further evaluate the frame-based patterns against state-of-the-art dependency and lexico-syntactic patterns. The results show that frame-based patterns perform comparatively better for smaller datasets and achieves a statistically higher performance for large datasets. The remainder of this paper is structured as follows: In Sect. 2, the related work for this study is described. In Sect. 3, we describe the proposed frame-based semantic patterns. In Sect. 4, we describe the datasets, evaluation metrics and the results of this study. Finally, in Sect. 5, we conclude this paper.

2 Related Work

The use of frames for information extraction (IE) was first investigated as early as 1995 [12] to automatically build a knowledge base of domain-specific linguistic patterns. Following this, there have been very few subsequent studies that have further explored the use of frames for IE. A major bottleneck to this is the absence of tools that can handle the complexity of annotating sentences with frames. However, the recent development of frame-based linguistic resources such as FrameNet [3, 8], and FrameNet based semantic parsing tools such as SEMAFOR [6], has reinstated interest in using frames for IE. FrameNet annotations has been successfully used in various application domains [13, 17]. On the other hand, over the years the field of relation extraction has witnessed significant amount of research. [14] identify at least three learning paradigms in the field of relation extraction, which include (a) supervised approaches focussing on hand-labelled datasets [18]; (b) unsupervised approaches targeting large amounts of text [4]; and (c) bootstrapping method that starts with small seed instances to iteratively learn patterns and entity pairs [1, 5, 15]. Further [14] proposed distant supervision method employing Freebase to automatically develop a large dataset

¹ The online demo of SEMAFOR is available at <http://demo.ark.cs.cmu.edu/parse>.

from Wikipedia text to exploit relation extraction. [16] proposed to relax the distant supervision method and used mutually exclusive training knowledge base and training text to develop a different dataset for relation extraction.

In relation to the above studies, the present study examines frame-based semantic patterns for pattern-based relation extraction. Although several studies have employed frames for different IE tasks, to the best knowledge of the authors, there are no previous studies that specifically examine frames for pattern-based relation extraction. The proposed frame-based patterns are evaluated on different datasets developed following distant supervision method. The focus of this study is not to develop a classifier that competes with other relation extraction systems that uses the above datasets. However, this study specifically evaluates the usefulness of frame-based semantic patterns for relation extraction. To this end, the proposed frame-based patterns are compared with dependency and lexico-syntactic pattern types, which are the state-of-the-art for pattern-based relation extraction.

3 Frame-Based Semantic Patterns

We present in this section two types of frame-based semantic patterns, based on FrameNet annotations: (i) *Frame Element* patterns and (ii) *Frame* patterns. To aid understanding we commence the section by briefly describing FrameNet, and then explain the proposed patterns.

3.1 FrameNet

FrameNet is a lexical resource primarily designed to support natural language processing. Founded on the theory of frame semantics [7], FrameNet comprises a large collection of *frames*, each identified by a label, and each describing some “happening” (situation). Each *Frame* comprises *Frame Elements* (FEs) describing semantic roles within the context of the situation described by the frame. For example, the `Commercial_sell` frame describes basic commercial transactions involving buyers and sellers, exchanging money and goods. The FEs in this case are `Buyer`, `Seller`, and `Goods`. Words that *evoke* these frames are called Lexical Units (LUs). An example sentence with FrameNet annotations was earlier provided in Sect. 1. The FrameNet annotations are used to develop the following two types of patterns (a) *Frame Element patterns* and (b) *Frame patterns*.

3.2 Frame Element Patterns

The *Frame Element* pattern is developed based on the mapping provided between *frames* and *frame element* annotations provided by FrameNet. For example, consider the FrameNet annotations for the sentence “Currently, he works at Twitter in San Francisco” provided in Fig. 1.

The FrameNet annotations provided for this sentence include *frames* such as `Being Employed` and `Businesses` (marked in blue) and the *frame elements*



Fig. 1. Frame annotation for example sentence. (Color figure online)

Place of Employment, Business and Place, marked in red. The mapping between these *frames* and *frame elements* can be used to create the following patterns to extract the entities (*Twitter*, *San Francisco*) for the COMPANY-LOCATION relation: (1) “BUSINESSES BUSINESS PLACE”; (2) “BEING_EMPLOYED PLACE_OF_EMPLOYMENT”.

3.3 Frame Patterns

The *frame element* pattern described above provides a very specific mapping between related entities. However, in many instances such fine grained annotations are not available. In such case, the *Frame* pattern is proposed to map map entities for a given relation. For example, consider the frame annotations for the example sentence shown in Fig. 2. From the figure it can be seen that although various FEs are triggered by the LUs Microsoft Corporation, Redmond and Washington relevant to the COMPANY-LOCATION relation of interest, there is no individual frame that can be used to define a pattern to map entities with respect to this relation. Instead the *frame* names (shown in blue) can be used to define patterns for relation extraction as follows: (1) “BUSINESSES MEMBERSHIP BUSINESSES POLITICAL_LOCALES”; (2) “BUSINESSES MEMBERSHIP BUSINESSES POLITICAL_LOCALES LOCATION” to extract COMPANY-LOCATION(*Microsoft Corporation*, *Redmond*) and COMPANY-LOCATION(*Microsoft Corporation*, *Washington*). Such Frame patterns also provides the additional advantage that they are more general than *frame element* patterns.



Fig. 2. Frames annotation for example sentence. (Color figure online)

4 Evaluation

In this section, we describe the lexico-syntactic and dependency based syntactic patterns against which frame-based patterns are evaluated. The datasets, evaluation metrics and the results of this study is also presented in this section.

4.1 Patterns Evaluated Against

The proposed frame-based patterns (F) are evaluated against the following pattern types:

a. Lexico-syntactic patterns. F is evaluated against the following two types of lexico-syntactic patterns: (a) L1 - patterns using lexical entries between entities; and (b) L2 - patterns replacing lexical entries with their Part Of Speech (POS) tags

b. Dependency-based syntactic patterns. The shortest path between entities in the dependency tree is shown to be useful for relation extraction [19]. F is evaluated against the following three types of dependency-based syntactic patterns based on the shortest path: (a) D1 - patterns using lexical entries between entities; (b) D2 - patterns using dependency relations between entities; and (c) D3 - patterns using both lexical entries and grammatical relations between entities.

4.2 Datasets

The proposed frame-based patterns are evaluated on the following three datasets.

SemEval-2010 Task 8 Dataset. The SemEval-2010 Task 8 dataset [10] is a standard dataset for relation extraction, containing 10,717 examples annotated with 9 different relation types, and an artificial relation ‘Other’. The nine relations are: CAUSE-EFFECT, COMPONENT-WHOLE, CONTENT-CONTAINER, ENTITY-DESTINATION, ENTITY-ORIGIN, INSTRUMENT-AGENCY, MEMBER-COLLECTION, MESSAGE-TOPIC and PRODUCT- PRODUCER. The dataset is split into 8,000 training examples and 2,717 test examples. The dataset is split into 8,000 training examples and 2,717 test examples, with each sentence marked with two nominals, e_1 and e_2 . The task is to predict the relation between the nominals considering the directionality. Thus, the relation Cause-Effect(e_1, e_2) is different from the relation Cause-Effect(e_2, e_1). Considering directionality results in 18 different relations. However, only 17 relations were used since one of the relations had only one annotated instance. The instances are randomly split in the ratio 80:20 to create the training and the test set, respectively.

[16] Dataset. The [16] dataset was developed with a focus to relax the distant supervision assumption to extract relations from newswire instead of Wikipedia. In this study, we considered the ten relations shown in Table 1 from Riedel et al. (2010) dataset to evaluate frame-based patterns. Sentences for each of these relations were randomly split in the ration of 80:20 to create the training and test set, respectively.

Table 1. Relations considered from [16] dataset; TS: Total Sentences

	Relation	TS
REL_1	people_deceased_person_place_of_death	2541
REL_2	people_person_place_of_birth	4265
REL_3	business_person_company	7987
REL_4	location_administrative_division_country	8860
REL_5	location_country_administrative_divisions	8860
REL_6	location_neighborhood_neighborhood_of	9472
REL_7	people_person_place_lived	9829
REL_8	location_country_capital	11216
REL_9	people_person_nationality	11446
REL_10	location_location_contains	75969
Total number of sentences: 150445		

Wikipedia dataset. The Wikipedia dataset was specifically developed for this study, following the distant supervision method. Specifically, we find all sentences that mentions a pair of entities in the seed dataset, and consider those sentences as describing the semantic relationship between the two entities specified in the seed dataset. DBpedia [2] was used to obtain seed entity pairs for ten different relations, which were further used to obtain sentences from Wikipedia dump. Sentences with a mention of at least one entity pair was retained. The dataset was randomly split in the ratio of 80:20 to create the training and the test set, respectively. The number of sentences extracted for different relations are as follows: (1) ACTOR-MOVIE-3147; (2) COMPANY-LOCATION-6908; (3) COMPANY-PRODUCT-9122; (4) DIRECTOR-MOVIE-10651; (5) AUTHOR-BOOKTITLE-12245; (6) COMPANY-FOUNDER-14489; (7) ALBUM-ARTIST-20961; (8) BIRTHPLACE-PERSON-21737; (9) ALBUM-GENRE-22934; and (10) COUNTRY-CITY-45981.

4.3 Evaluation Metrics

As previously discussed in Sect. 1, the evaluation of a pattern set, besides considering its ability to match test sentences, should also examine how correctly, the patterns extract related entities for a given relation. The precision and recall measures employed in this study are designed to include this aspect. Thus, given a pattern $l \in \mathcal{L}$, the pattern set obtained from train data, and a test sentence $s \in \mathcal{S}$, the following types of patterns are defined:

(a) *matched pattern*: the pattern l is defined as a *matched pattern* for the test sentence s , iff (if and only if) the pattern l matches the test sentence s .

(b) *correct pattern*: a pattern l is defined as a *correct pattern* for the test sentence s , iff the pattern l matches the test sentence s and correctly extracts the two arguments (e_1, e_2) for a given relation r .

The precision of a pattern l is defined as the ratio of number of times the pattern l is seen as a *correct pattern* to the number of times it is seen as a matched pattern on the test set \mathcal{S} . Thus, the precision of a pattern l on the test set \mathcal{S} is given by:

$$\text{Precision}(l) = \frac{\# \text{ pattern } l \text{ is a } \textit{correct pattern} \text{ in } \mathcal{S}}{\# \text{ pattern } l \text{ is a } \textit{matched pattern} \text{ in } \mathcal{S}}.$$

The overall precision P of the pattern set is obtained by:

$$P = \frac{1}{|\mathcal{L}|} \sum_{l \in \mathcal{L}} \text{Precision}(l),$$

where $|\mathcal{L}|$ is the total number of patterns in the pattern set.

The recall of a pattern set is measured in terms of its effectiveness or coverage in applying correct patterns on the test set and is defined as the ratio of the total number of test sentences on which *correct patterns* are applied to the total number of test sentences. Thus, the recall R of a pattern set is given by:

$$R = \frac{\# \text{ of test sentences with } \textit{correct patterns}}{\# \text{ of test sentences}}.$$

Given Precision P and Recall R , the F-score of a pattern set is obtained by:
F-Score = $\frac{2 \times PR}{P+R}$.

4.4 Results

The following are the evaluation results.

Pattern sets applied in isolation. The F-score values of various pattern sets (L1, L2, D1, D2, D3, F), when applied in isolation on three different datasets for relation extraction are shown in Tables 2, 3 and 4. As seen, the frame-based pattern set (F) perform comparatively better for Semeval 2010 Task 8 dataset (Table 2) achieving an average F-score of 0.64, against other pattern types. With respect to the Wikipedia and Riedel et al. [16] datasets (Tables 3 and 4), F achieves a statistically significant F-score of 0.66 and 0.76 ($p \leq 0.05$; Wilcoxon Signed-Rank Test) against other pattern types.

Frames vs. dependency patterns. As seen in Tables 2, 3 and 4 the dependency-based patterns using grammatical relations (D2) achieves the second best performance among the different patterns evaluated. To evaluate the performance of D2 against F, we compare their precision and recall scores for relations in Wikipedia dataset (results are presented only for Wikipedia dataset due to space constraints) in Table 5. As seen in Table 5, F achieves statistically significant precision and recall in comparison to the scores achieved by D2

Table 2. F-score values for relations in Semeval 2010 Task 8 dataset. † statistically significant against F.

Relation	Patterns applied in isolation						Augmented Patterns		
	L1	L2	D1	D2	D3	F	FAD	FAL	LAD
	F-score								
CAUSE_EFFECT_e1_e2	0.34	0.42	0.60	0.62	0.59	0.64	0.66	0.68	0.60
CAUSE_EFFECT_e2_e1	0.41	0.52	0.49	0.56	0.56	0.59	0.62	0.67	0.58
COMPONENT_WHOLE_e1_e2	0.62	0.64	0.19	0.62	0.62	0.70	0.64	0.74	0.58
COMPONENT_WHOLE_e2_e1	0.48	0.62	0.45	0.71	0.70	0.66	0.70	0.72	0.73
CONTENT_CONTAINER_e1_e2	0.38	0.50	0.44	0.59	0.49	0.55	0.63	0.58	0.57
CONTENT_CONTAINER_e2_e1	0.54	0.63	0.40	0.65	0.57	0.67	0.73	0.72	0.72
ENTITY_DESTINATION_e1_e2	0.27	0.27	0.70	0.58	0.69	0.62	0.67	0.71	0.48
ENTITY_ORIGIN_e1_e2	0.32	0.59	0.59	0.63	0.58	0.68	0.68	0.74	0.66
ENTITY_ORIGIN_e2_e1	0.74	0.79	0.28	0.79	0.68	0.87	0.85	0.76	0.78
INSTRUMENT_AGENCY_e1_e2	0.42	0.52	0.39	0.54	0.53	0.72	0.75	0.67	0.57
INSTRUMENT_AGENCY_e2_e1	0.23	0.41	0.39	0.69	0.42	0.44	0.56	0.52	0.69
MEMBER_COLLECTION_e1_e2	0.50	0.50	0.42	0.66	0.56	0.65	0.71	0.69	0.70
MEMBER_COLLECTION_e2_e1	0.63	0.83	0.37	0.75	0.72	0.87	0.78	0.88	0.76
MESSAGE_TOPIC_e1_e2	0.14	0.43	0.43	0.53	0.37	0.57	0.63	0.65	0.59
MESSAGE_TOPIC_e2_e1	0.01	0.27	0.59	0.57	0.44	0.33	0.63	0.38	0.60
PRODUCT_PRODUCER_e1_e2	0.39	0.65	0.41	0.69	0.58	0.72	0.78	0.81	0.74
PRODUCT_PRODUCER_e2_e1	0.14	0.51	0.62	0.68	0.61	0.71	0.74	0.77	0.65
AVERAGE	0.38	0.53	0.45	0.63	0.57	0.64	0.69 †	0.68 †	0.64

($p \leq 0.05$; Wilcoxon Signed-Rank Test). As explained previously (Sect. 4.3), the precision metric considered both to *match* test sentences and extract *correct* entities. Thus, the precision scores (Table 5) shows that F patterns are more precise than D2. Similarly, the recall metric measured the coverage of applying *correct patterns* on test sentences. The recall scores in Table 5 shows that F patterns apply more *correct patterns* as against D2 patterns.

Frame element patterns vs. frame patterns. The precision and recall scores achieved by *frame element* patterns and *frame* patterns (described in Sect. 3) individually are presented in Table 6. As seen in Table 6, both *frame element* and *frame* patterns achieve a similar precision score. However, in terms of recall, the *frame element* patterns have a lower coverage (0.26) as against *frame* patterns (0.57). This shows that *frame element* patterns apply *correct patterns* on lesser number of test sentences as against *frame* patterns.

Table 3. F-score values for relations in Wikipedia dataset. * statistically significant against L1, L2, D1, D2, D3. ** statistically significant against LAD. † statistically significant against F.

Relation	Patterns applied in isolation						Augmented Patterns		
	L1	L2	D1	D2	D3	F	FAD	FAL	LAD
	F-score								
ACTOR-MOVIE	0.27	0.48	0.34	0.57	0.65	0.67	0.72	0.71	0.66
COMPANY-LOCATION	0.37	0.51	0.36	0.59	0.38	0.68	0.73	0.70	0.65
COMPANY-PRODUCT	0.22	0.40	0.40	0.61	0.36	0.69	0.76	0.72	0.66
DIRECTOR-MOVIE	0.23	0.30	0.56	0.65	0.60	0.71	0.75	0.73	0.70
AUTHOR-BOOKTITLE	0.26	0.49	0.48	0.63	0.52	0.69	0.74	0.73	0.68
COMPANY-FOUNDER	0.45	0.61	0.41	0.72	0.80	0.72	0.82	0.80	0.78
ALBUM-ARTIST	0.27	0.50	0.48	0.60	0.47	0.64	0.72	0.71	0.66
BIRTHPLACE-PERSON	0.27	0.45	0.24	0.53	0.28	0.56	0.65	0.65	0.6
ALBUM-GENRE	0.33	0.41	0.42	0.56	0.49	0.65	0.68	0.71	0.63
COUNTRY-CITY	0.42	0.55	0.33	0.64	0.42	0.68	0.75	0.72	0.70
AVERAGE	0.30	0.47	0.40	0.61	0.49	0.66*	0.73**†	0.71**†	0.67

Table 4. F-score values for relations in [16] dataset. * statistically significant against L1, L2, D1, D2, D3. ** statistically significant against LAD. † statistically significant against F.

Relation	Patterns applied in isolation						Augmented Patterns		
	L1	L2	D1	D2	D3	F	FAD	FAL	LAD
	F-score								
people_deceased_person_place_of_death	0.21	0.36	0.26	0.35	0.28	0.68	0.63	0.61	0.52
people_person_place_of_birth	0.37	0.44	0.52	0.57	0.50	0.78	0.78	0.52	0.67
business_person_company	0.42	0.25	0.64	0.54	0.51	0.88	0.91	0.89	0.71
location_administrative_division_country	0.60	0.63	0.52	0.59	0.50	0.84	0.93	0.82	0.82
location_country_administrative_divisions	0.59	0.65	0.19	0.28	0.12	0.70	0.67	0.72	0.69
location_neighborhood_neighborhood_of	0.85	0.86	0.33	0.67	0.66	0.79	0.93	0.93	0.85
people_person_place_lived	0.53	0.59	0.44	0.67	0.55	0.82	0.83	0.82	0.75
location_country_capital	0.53	0.55	0.11	0.19	0.19	0.61	0.50	0.60	0.59
people_person_nationality	0.65	0.69	0.45	0.70	0.61	0.82	0.86	0.85	0.81
AVERAGE	0.52	0.55	0.37	0.51	0.43	0.76*	0.78**†	0.75**†	0.71

Wikipedia vs. Riedel et al. [16] dataset. As seen from Tables 3 and 4, the performance of F is significantly higher for Riedel dataset in comparison to Wikipedia dataset. While F achieves an average F-score of 0.66 as against an average F-score of 0.61 obtained by D2 for Wikipedia dataset, F achieves a higher average F-score of 0.76 against the average F-score of 0.51 achieved by D2 for the Riedel dataset. These results are significant since the two datasets are developed following different methods. While the Wikipedia dataset follows distant supervision method, the Riedel dataset is developed by relaxing the

Table 5. Precision and recall values of D2 and F pattern sets for relations in Wikipedia dataset. * statistically significant.

Relation	D2-P	F-P	D2-R	F-R
ACTOR-MOVIE	0.60	0.67	0.55	0.69
COMPANY-LOCATION	0.62	0.74	0.58	0.64
COMPANY-PRODUCT	0.76	0.79	0.52	0.62
DIRECTOR-MOVIE	0.63	0.73	0.69	0.70
AUTHOR-BOOKTITLE	0.64	0.73	0.63	0.66
COMPANY-FOUNDER	0.86	0.88	0.62	0.80
ALBUM-ARTIST	0.60	0.64	0.62	0.64
BIRTHPLACE-PERSON	0.51	0.54	0.56	0.60
ALBUM-GENRE	0.50	0.58	0.64	0.74
COUNTRY-CITY	0.70	0.70	0.60	0.67
AVERAGE	0.64	0.70*	0.60	0.67*

distant supervision method. As seen from these Tables (3 and 4), the performance of D2 decreases for Riedel dataset in comparison to Wikipedia dataset (0.61 vs. 0.51), showing that dependencies are less useful for datasets where mutually exclusive training knowledge base and training text is employed for relation extraction. On the other hand, the performance of F is higher for Riedel dataset, indicating that frame-based patterns can generalize well for such datasets.

Augmenting dependency patterns and lexico-syntactic patterns with frames. The results of augmenting dependency and lexical patterns with frames is also provided in Tables 2, 3 and 4. A cascaded method was followed to sequentially apply one pattern set after another on the test set. The patterns from the second pattern set was applied on those test sentences, where the first pattern set failed to apply *correct patterns* (i.e., not only *match* but also apply *correct* patterns). Accordingly, two types of frame-based augmented patterns were examined: FAD - frame (F) augmented dependency (D2) patterns; and FAL - frame (F) augmented lexical (L2) patterns. These patterns were evaluated against dependency patterns (D2) augmented with lexical patterns (L2). The D2 and L2 were chosen as these patterns achieved higher performance among related pattern types. The F-score values of FAD and FAL in all three datasets (Tables 2, 3 and 4) shows that FAD and FAL achieve statistically significant performance against using frame-based patterns (F) in isolation. Interestingly, FAD and FAL achieves a higher performance against combining dependencies and lexical patterns. This indicates that augmenting dependency and lexical patterns with frames are useful for relation extraction. The precision and recall values (not shown here) of augmented patterns indicates that augmented patterns achieve higher recall, thus applying *correct patterns* on larger number of test sentences.

Table 6. Precision and recall values of *frame element* and *frame* patterns.

Relation	<i>Frame element</i>		<i>Frame</i>	
	Patterns		Patterns	
	P	R	P	R
ACTOR-MOVIE	0.60	0.63	0.72	0.17
COMPANY-LOCATION	0.69	0.53	0.70	0.32
COMPANY-PRODUCT	0.74	0.48	0.75	0.34
DIRECTOR-MOVIE	0.70	0.64	0.66	0.17
AUTHOR-BOOKTITLE	0.68	0.60	0.64	0.24
COMPANY-FOUNDER	0.80	0.53	0.78	0.32
ALBUM-ARTIST	0.59	0.53	0.64	0.32
BIRTHPLACE-PERSON	0.51	0.52	0.51	0.22
ALBUM-GENRE	0.56	0.66	0.55	0.23
COUNTRY-CITY	0.66	0.58	0.65	0.35
AVERAGE	0.65	0.57	0.66	0.26

5 Conclusion

To conclude, we presented in this paper frame-based semantic patterns for relation extraction. More specifically, we proposed *frame element* and *frame* patterns exploiting the FrameNet annotations for relation extraction, which were evaluated against lexico-syntactic and dependency-based syntactic patterns, on three independent datasets. The evaluation results shows that frame-based patterns achieves significantly higher performance against state-of-the-art dependency and lexical patterns, both in terms of precision and recall on all three datasets. Experiments conducted to augment dependency and lexical patterns with frame-based patterns shows that the augmentation helps in achieving higher recall.

References

1. Agichtein, E., Gravano, L.: Snowball: extracting relations from large plain-text collections. In: Proceedings of the Fifth ACM Conference on Digital Libraries, pp. 85–94. ACM (2000)
2. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Ives, Z.: Dbpedia: a nucleus for a web of open data. In: 6th International Semantic Web Conference, Busan, Korea (2007)
3. Baker, C.F., Fillmore, C.J., Lowe, J.B.: The berkeley framenet project. In: Proceedings of the 17th International Conference on Computational Linguistics, vol. 1, pp. 86–90. Association for Computational Linguistics (1998)
4. Banko, M., Cafarella, M.J., Soderland, S., Broadhead, M., Etzioni, O.: Open information extraction for the web. In: Proceedings of IJCAI, vol. 7, pp. 2670–2676 (2007)

5. Brin, S.: Extracting patterns and relations from the world wide web. In: Atzeni, P., Mendelzon, A., Mecca, G. (eds.) WebDB 1998. LNCS, vol. 1590, pp. 172–183. Springer, Heidelberg (1999). https://doi.org/10.1007/10704656_11
6. Das, D., Chen, D., Martins, A.F., Schneider, N., Smith, N.A.: Frame-semantic parsing. *Comput. Linguist.* **40**(1), 9–56 (2014)
7. Fillmore, C.: Frame semantics. In: *Linguistics in the Morning Calm*, pp. 111–137 (1982)
8. Fillmore, C.J., Johnson, C.R., Petruck, M.R.: Background to framenet. *Int. J. Lexicogr.* **16**(3), 235–250 (2003)
9. Hearst, M.A.: Automatic acquisition of hyponyms from large text corpora. In: *Proceedings of the COLING*, pp. 539–545. Association for Computational Linguistics (1992)
10. Hendrickx, I., Kim, S.N., Kozareva, Z., Nakov, P., Ó Séaghdha, D., Padó, S., Pennacchiotti, M., Romano, L., Szpakowicz, S.: Semeval-2010 task 8: multi-way classification of semantic relations between pairs of nominals. In: *Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions*, pp. 94–99. Association for Computational Linguistics (2009)
11. Jijkoun, V., De Rijke, M., Mur, J.: Information extraction for question answering: improving recall through syntactic patterns. In: *Proceedings of the COLING*, p. 1284. Association for Computational Linguistics (2004)
12. Kim, J.T., Moldovan, D., et al.: Acquisition of linguistic patterns for knowledge-based information extraction. *IEEE Trans. Knowl. Data Eng.* **7**(5), 713–724 (1995)
13. Liu, S., Chen, Y., He, S., Liu, K., Zhao, J.: Leveraging framenet to improve automatic event detection. In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, vol. 1, pp. 2134–2143. Association for Computational Linguistics, Berlin, August 2016. Long Papers. <http://www.aclweb.org/anthology/P16-1201>
14. Mintz, M., Bills, S., Snow, R., Jurafsky, D.: Distant supervision for relation extraction without labeled data. 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*, vol. 2, pp. 1003–1011. Association for Computational Linguistics (2009)
15. Ravichandran, D., Hovy, E.: Learning surface text patterns for a question answering system. In: *Proceedings of the COLING*, pp. 41–47. Association for Computational Linguistics (2002)
16. Riedel, S., Yao, L., McCallum, A.: Modeling relations and their mentions without labeled text. In: Balcázar, J.L., Bonchi, F., Gionis, A., Sebag, M. (eds.) *ECML PKDD 2010*. LNCS (LNAI), vol. 6323, pp. 148–163. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15939-8_10
17. Søgaard, A., Plank, B., Alonso, H.M.: Using frame semantics for knowledge extraction from twitter. In: *Proceedings of AAAI* (2015)
18. Surdeanu, M., Ciaramita, M.: Robust information extraction with perceptrons. In: *Proceedings of the NIST 2007 Automatic Content Extraction Workshop (ACE07)* (2007)
19. Wu, F., Weld, D.S.: Open information extraction using wikipedia. In: *Proceedings of COLING*, pp. 118–127. Association for Computational Linguistics (2010)



Semantic Refinement GRU-Based Neural Language Generation for Spoken Dialogue Systems

Van-Khanh Tran^{1,2}(✉) and Le-Minh Nguyen¹

¹ Japan Advanced Institute of Science and Technology, JAIST, 1-1 Asahidai, Nomi, Ishikawa 923-1292, Japan

{[tvkhanh](mailto:tvkhanh@jaist.ac.jp),[nguyenml](mailto:nguyenml@jaist.ac.jp)}@jaist.ac.jp

² University of Information and Communication Technology, ICTU, Thai Nguyen University, Thai Nguyen, Vietnam
tvkhanh@ictu.edu.vn

Abstract. Natural language generation (NLG) plays a critical role in spoken dialogue systems. This paper presents a new approach to NLG by using recurrent neural networks (RNN), in which a gating mechanism is applied before RNN computation. This allows the proposed model to generate appropriate sentences. The RNN-based generator can be learned from unaligned data by jointly training sentence planning and surface realization to produce natural language responses. The model was extensively evaluated on four different NLG domains. The results show that the proposed generator achieved better performance on all the NLG domains compared to previous generators.

1 Introduction

Natural Language Generation is a critical component in a Spoken Dialogue System (SDS), and its task is to convert a meaning representation produced by the dialogue manager into natural language utterances. Conventional approaches to NLG follow a pipeline which typically breaks down the task into sentence planning and surface realization. Sentence planning decides the order and structure of sentence, which is followed by a surface realization which converts the sentence structure into the final utterance. Previous approaches to NLG still rely on extensive hand-tuning templates and rules that require expert knowledge of linguistic representation. There are some common and widely approaches to solve NLG problems, including rule-based [3], corpus-based n-gram models [10], and a trainable generator [15]. Joint based generators use a two-step pipeline [4, 14]; or applying a joint model for both tasks [19, 22].

Recently, approaches based on recurrent neural networks have shown advantages in solving the NLG tasks. RNN-based models have been used for NLG as a joint training model [19, 22] and an end-to-end training model [23]. A recurring problem in such systems is requiring datasets annotated for specific dialogue

acts¹ (DA). Moreover, previous works may lack ability to handle cases such as the binary slots (*i.e.*, *yes* and *no*) and slots that take *don't_care* value which cannot be directly delexicalized [19], or to generalize to unseen domains [22]. Furthermore, a problem of the current generators is that the generators produce the next token based on the information from the forward context, whereas the sentence may depend on the backward context. As a result, the generators tend to generate nonsensical utterances.

We propose a statistical NLG based on a gating mechanism on a GRU model, in which the gating mechanism is applied before RNN computation. The proposed model can learn from unaligned data by jointly training the sentence planning and surface realization to generate required sentences. We found that the proposed model can produce sentences in a more correct order than the existing models. The previous RNN-based generators may have lack of consistency about the order of slot-value pairs during generation. For example, given a DA with pattern: *Compare*(name = *A*, property1 = *a1*, property2 = *a2*, name = *B*, property1 = *b1*, property2 = *b2*). The pattern for correct utterances can be: [*A-a1-a2*, *B-b1-b2*], [*A-a2-a1*, *B-b2-b1*], [*B-b1-b2*, *A-a1-a2*], [*B-b1-b2*, *A-a2-a1*]. Therefore, a generated utterance: “The *A* has *a1* and ***b1*** properties, while the *B* has ***a2*** and *b2* properties” is an incorrect utterance, in which ***b1*** and ***a2*** properties were generated in wrong order. This occasionally leads to inadequate sentences.

We assessed the proposed generators on varied NLG domains, in which the results showed that our proposed method outperforms the previous methods in terms of BLEU [11] and slot error rate ERR [22] scores. To summary, we make three contributions in this study where we: (i) propose two semantic refinement RNN-based models, in which a gating mechanism is applied before computational RNN to refine the original inputs, (ii) conduct extensively experiments on four NLG domains, and (iii) analyze the effectiveness of the proposed models on ability to handle the undelexicalized tokens, and to generalize to unseen domain when limited amount of in-domain data was fed.

2 Related Work

Conventional approaches to NLG traditionally split the task into two subtasks: sentence planning and surface realization. Sentence planning deals with mapping of the input semantic symbols onto a linguistic structure, *e.g.*, a tree-like or a template structure. The surface realization then converts the structure into an appropriate sentence [15]. Despite their success and wide use in solving NLG problems, these traditional methods still rely on the handcrafted rule-based generators or rerankers. The authors in [10] proposed a class-based n-gram language model (LM) generator which can learn to generate the sentences for a given DA and then select the best sentences using a rule-based reranker. Some of the limitation of the class-based LMs were addressed in [13] by proposing a method

¹ A combination of an action type and a list of slot-value pairs. *e.g.* *inform*(name = ‘*Frances*’; area = ‘*City Center*’).

based on a syntactic dependency tree. A phrase-based generator based on factored LMs was introduced in [8], which can learn from a semantically aligned corpus.

Recently, RNNs-based approaches have shown promising performance in the NLG domain. The authors in [6, 17] used RNNs in a multi-modal setting to generate captions for images, while a generator using RNNs to create Chinese poetry was also proposed in [24]. The authors in [7] encoded an unstructured textual knowledge source along with previous responses and context to produce a response for technical support queries. For task-oriented dialogue systems, a combination of a forward RNN generator, a CNN reranker, and a backward RNN reranker was proposed in [19] to generate utterances. A semantically conditioned-based Long Short-Term Memory (LSTM) generator was introduced in [22], which proposed a control “reading” gate to the traditional LSTM cell and can learn the gating mechanism and language model jointly. A recurring problem in such systems is the lack of sufficient domain-specific annotated data.

3 Recurrent Neural Language Generator

The recurrent language generator proposed in this paper based on a RNN language model [9], which consists of three layers: an input layer, a hidden layer and an output layer. The input to the network at each time step t is a 1-hot encoding \mathbf{w}_t of a token² w_t which is conditioned on a recurrent hidden layer \mathbf{h}_t . The output layer \mathbf{y}_t represents the probability distribution of the next token given previous token w_t and hidden \mathbf{h}_t . We can sample from this conditional distribution to obtain the next token in a generated string, and feed it as the next input to the generator. This process finishes when a stop sign is generated [6], or some constraint are reached [24]. The network can generate a sequence of tokens which can be lexicalized³ to form the required utterance. Moreover, in order to ensure that the generated utterance represents the intended meaning of the given DA, the generator is further conditioned on a vector \mathbf{z} , a 1-hot vector representation of DA. Inspired by work in [18], we propose an intuition: *Gating before computation*, in which we add gating mechanism before the RNN computation to semantically refine the input tokens. The following sections present two proposed Semantic Refinement (SR) gating based RNN generators.

3.1 SRGRU-BASE

In this model, instead of feeding an input token \mathbf{w}_t to the RNN model at each time step t , the input token is filtered by a semantic gate which is computed as follows:

$$\begin{aligned} \mathbf{d}_t &= \sigma(\mathbf{W}_{dz}\mathbf{z}) \\ \mathbf{x}_t &= \mathbf{d}_t \odot \mathbf{w}_t \end{aligned} \tag{1}$$

² Input texts are delexicalized in which slot values are replaced by its corresponding slot tokens.

³ The process in which slot token is replaced by its value.

where: \mathbf{W}_{dz} is a trained matrix to project the given DA representation into the word embedding space, and \mathbf{x}_t is new input. Here \mathbf{W}_{dz} plays a role of sentence planning since it can directly capture which DA features are useful during the generation to encode the input information. The \odot element-wise multiplication plays a part in word-level matching which learns not only the vector similarity, but also preserve information about the two vectors. \mathbf{d}_t is called a *refinement gate* since the input tokens are refined by the DA information. As a result, we can represent the whole input sentence based on these refined inputs using RNN model.

In this study, we use GRU, which was recently proposed in [2], instead of LSTM as building computational block for RNN, which is formulated as follows:

$$\mathbf{r}_t = \sigma(\mathbf{W}_{rx}\mathbf{x}_t + \mathbf{W}_{rh}\mathbf{h}_{t-1}) \quad (2)$$

$$\mathbf{u}_t = \sigma(\mathbf{W}_{ux}\mathbf{x}_t + \mathbf{W}_{uh}\mathbf{h}_{t-1}) \quad (3)$$

$$\tilde{\mathbf{h}}_t = \tanh(\mathbf{W}_{hx}\mathbf{x}_t + r_t \odot \mathbf{W}_{hh}\mathbf{h}_{t-1}) \quad (4)$$

$$\mathbf{h}_t = \mathbf{u}_t \odot \mathbf{h}_{t-1} + (1 - \mathbf{u}_t) \odot \tilde{\mathbf{h}}_t \quad (5)$$

where: $\mathbf{W}_{rx}, \mathbf{W}_{rh}, \mathbf{W}_{ux}, \mathbf{W}_{uh}, \mathbf{W}_{hx}, \mathbf{W}_{hh}$ are weight matrices; $\mathbf{r}_t, \mathbf{u}_t$ are reset and update gate, respectively, and \odot denotes for element-wise product. The Semantic Refinement GRU (SRGRU)-Base architecture is shown in Fig. 1.

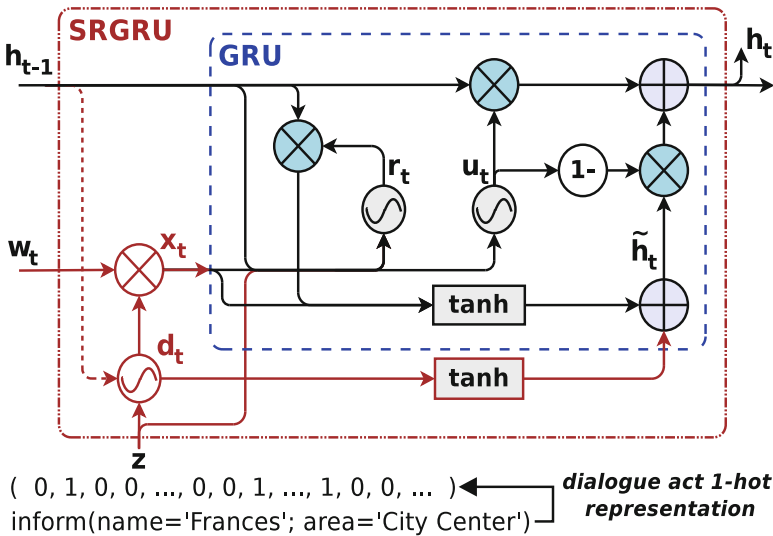


Fig. 1. SRGRU-Context cell. The blue dashed box is a traditional GRU cell in charge of surface realization, while the red parts form sentence planning based on a sigmoid control gate \mathbf{d}_t and a dialogue act \mathbf{z} . The contextual information \mathbf{h}_{t-1} is imported into the refinement gate \mathbf{d}_t via red dotted line. The SRGRU-Base is achieved by omitting this link. (Color figure online)

Finally, the output distribution of each token is defined by applying a softmax function g as follows:

$$P(w_{t+1} | w_t, w_{t-1}, \dots, w_0, \mathbf{z}) = g(\mathbf{W}_{ho}\mathbf{h}_t) \quad (6)$$

where: \mathbf{W}_{ho} is learned linear projection matrix. At training time, we use the ground truth token for the previous time step in place of the predicted output. At test time, we implement a simple beam search to over-generate several candidate responses.

3.2 SRGRU-CONTEXT

SRGRU-Base uses only the DA information to gate the input sequence token by token. As a results, this gating mechanism may not capture the relationship between multiple words. In order to import context information into the gating mechanism, the Eq. 1 is modified as follows:

$$\begin{aligned} \mathbf{d}_t &= \sigma(\mathbf{W}_{dz}\mathbf{z} + \mathbf{W}_{dh}\mathbf{h}_{t-1}) \\ \mathbf{x}_t &= \mathbf{d}_t \odot \mathbf{w}_t \end{aligned} \quad (7)$$

where: \mathbf{W}_{dz} and \mathbf{W}_{dh} are weight matrices. \mathbf{W}_{dh} acts like a key phrase detector that learns to capture the pattern of generation tokens or the relationship between multiple tokens. In other words, the new input \mathbf{x}_t consists of information of the original input token \mathbf{w}_t , the dialogue act \mathbf{z} , and the hidden context \mathbf{h}_{t-1} . \mathbf{d}_t is called the *refinement* gate because the input tokens are refined by a combination gating information of the dialogue act \mathbf{z} and the previous hidden state \mathbf{h}_{t-1} . By taking advantage of gating mechanism from the LSTM model [5] in which the gating mechanism is employed to solve the gradient vanishing and exploding problem, we propose to apply the refinement gate deeper into the GRU cell. Firstly, the GRU reset and update gates can be further influenced on the given dialogue act \mathbf{z} and the refined input \mathbf{x}_t . The Eqs. (2) and (3) are modified as follows:

$$\mathbf{r}_t = \sigma(\mathbf{W}_{rx}\mathbf{x}_t + \mathbf{W}_{rh}\mathbf{h}_{t-1} + \mathbf{W}_{rz}\mathbf{z}) \quad (8)$$

$$\mathbf{u}_t = \sigma(\mathbf{W}_{ux}\mathbf{x}_t + \mathbf{W}_{uh}\mathbf{h}_{t-1} + \mathbf{W}_{uz}\mathbf{z}) \quad (9)$$

where: \mathbf{W}_{rz} and \mathbf{W}_{uz} act like background detectors that learn to control the style of the generating sentence. Secondly, Eq. (4) is modified so that the candidate activation $\tilde{\mathbf{h}}_t$ also depends on the refinement gate,

$$\tilde{\mathbf{h}}_t = \tanh(\mathbf{W}_{hx}\mathbf{x}_t + \mathbf{r}_t \odot \mathbf{W}_{hh}\mathbf{h}_{t-1}) + \tanh(\mathbf{W}_{dc}\mathbf{d}_t) \quad (10)$$

By this way, the reset and update gates learn not only the long-term dependency but also the gating information from the dialogue act and the previous hidden state. We call the resulting architecture Semantic Refinement GRU (SRGRU)-Context which is shown in Fig. 1.

3.3 Training

The cost function was the negative log-likelihood and computed by:

$$F(\theta) = - \sum_{t=1}^T \mathbf{y}_t^\top \log \mathbf{p}_t \quad (11)$$

where \mathbf{y}_t is the ground truth word distribution, \mathbf{p}_t is the predicted word distribution, T is length of the input sequence. The generators were trained by treating each sentence as a mini-batch with the l_2 regularization was added to the cost function for every 10 training examples. The models were initialized with pre-trained word vectors GLOVE [12] and optimized by using stochastic gradient descent and back propagation through time. To prevent over-fitting, early stopping was implemented using a validation set.

3.4 Decoding

The decoding phase we employ here is similar to [16] which consists of over-generation and re-ranking phases. The forward generator, in the over-generation phase, is conditioned on the given DA uses a beam search algorithm to generate candidate utterances, whereas the cost of forward generator $F_{fw}(\theta)$, in the re-ranking phase, is computed to form the re-ranking score R as follows:

$$R = F_{fw}(\theta) + \lambda ERR \quad (12)$$

where λ is a trade off constant which is set to a large value in order to severely penalize nonsensical outputs. The slot error rate ERR , which is the number of generated slots that are either redundant or missing, is computed by:

$$ERR = \frac{p + q}{N} \quad (13)$$

where N is the total number of slots in DA, and p, q is the number of missing and redundant slots, respectively. The ERR re-ranking criteria as mentioned in [22] cannot handle arbitrary slot-value pairs, *i.e.* *binary* slots or slots that take *don't_care* value, because such these pairs cannot be delexicalized and matched.

4 Experiments

4.1 Datasets

We conducted experiments using four different NLG domains: finding a restaurant, finding a hotel, buying a laptop, and buying a television. The Restaurant and Hotel domains were collected in [22] which contain system dialogue acts, shared slots, and specific domain slots. The Laptop and TV datasets have been released in [20] with about 13K distinct DAs in the Laptop and 7K distinct DAs in the TV. These two datasets have a much larger input space but only one training example for each DA so that the system must learn partial realization of concepts and be able to recombine and apply them to unseen DAs. The number of dialogue act types and slots of datasets is also larger than in Restaurant and Hotel datasets. As a result, the NLG tasks for the Laptop and TV datasets become much harder.

4.2 Experimental Setups

The generators were implemented using the TensorFlow library [1] and trained by partitioning each of the datasets into training, validation and testing set in the ratio 3:1:1. The hidden layer size was set to be 80, and the generators were trained with a 70% of dropout rate. We perform 5 runs with different random initialization of the network and the training is terminated by using early stopping as described in Sect. 3.3. We select model that yields the highest BLEU score on the validation set. The decoder procedure used beam search with a beam width of 10. We set λ to 1000 to severely discourage the reranker from selecting utterances which contain either redundant or missing slots. For each DA, we over-generated 20 candidate utterances and selected the top 5 realizations after reranking. Because the proposed models work stochastically, except the results reported in Table 1, all the results shown were averaged over 5 randomly initialized networks.

Since some sentences may depend on both the past and the future during generation, we train another backward SRGRU-Context to utilizing the flexibility of the refinement gate \mathbf{d}_t , in which we tie its weight matrices such \mathbf{W}_{dz} and \mathbf{W}_{dh} (Eq. 7) for both. We found that by tying matrix \mathbf{W}_{dz} for both forward and backward RNNs, the proposed generator seems to produce more correct and grammatical utterances than those having the only forward RNN. This model called Tying Backward SRGRU-Context (TB-SRGRU).

In order to better understand the effectiveness of the proposed model, we conduct more experiments to compare the SRGRU-Context with the previous generator SCLSTM in a variety of setups on proportion of training corpus, beam size, and top- k best results. Firstly, the Restaurant and TV datasets were chosen, in which the SCLSTM model obtained the best performances and the NLG task comes from a limited domain to a more diverse domain as described in Sect. 4.1. In this setup, the models were run with different size of training corpus. Secondly, we examined the stability of SRGRU-Context model on different setups of beam size and top- k best results.

4.3 Evaluation Metrics and Baselines

The generator performance was assessed by using two objective evaluation metrics, the BLEU score and the slot error rate ERR. Note that the slot error rate ERR was computed as an auxiliary metric alongside the BLEU score and calculated by averaging slot errors over each of the top 5 realizations in the entire corpus. Both metrics were computed by adopting code from an open source benchmark toolkit for Natural Language Generation⁴.

We compared our proposed models against with the general GRU (GRU-Base) and three strong baselines released from the NLG toolkit:

- ENCDEC proposed in [21] which applies the attention mechanism to an RNN encoder-decoder.

⁴ <https://github.com/shawnwun/RNNLG>.

- HLSTM proposed in [19] which uses a heuristic gate to ensure that all of the attribute-value information was accurately captured when generating.
- SCLSTM proposed in [22] which can learn the gating signal and language model jointly.

5 Results and Analysis

Overall, the proposed models SRGRUs consistently achieve better performance in term of the BLEU score in all domains. Especially, on the Hotel and TV datasets, the proposed models outperform the previous methods in both evaluation metrics. Moreover, our models also outperform the GRU basic model (GRU-Base) in all cases. However, the proposed models get worse on the Restaurant and Hotel datasets in terms of the error rate ERR score in comparison with SCLSTM. This indicates the advantage of the proposed refinement gate. A comparison of the two proposed generators is shown in Table 2: Without the backward RNN reranker, the generator tends to make semantic errors since it gains

Table 1. Comparison performance on four datasets in terms of the BLEU and the error rate ERR (%) scores; **bold** denotes the best and *italic* shows the second best model. The results were produced by training each network on 5 random initialization and selected model with the highest validation BLEU score.

Model	Restaurant		Hotel		Laptop		TV	
	BLEU	ERR	BLEU	ERR	BLEU	ERR	BLEU	ERR
ENCDEC	0.7398	2.78%	0.8549	4.69%	0.5108	4.04%	0.5182	3.18%
HLSTM	0.7466	0.74%	0.8504	2.67%	0.5134	1.10%	0.5250	2.50%
SCLSTM	0.7525	0.38%	0.8482	3.07%	0.5116	0.79%	0.5265	2.31%
GRU-Base	0.7381	1.41%	0.8455	2.66%	0.5153	1.77%	0.5245	2.03%
SRGRU-Base	0.7549	0.56%	0.8640	<i>1.21%</i>	0.5190	1.56%	0.5305	1.62%
SRGRU-Context	0.7634	0.49%	0.8776	0.98%	<i>0.5191</i>	1.19%	0.5311	1.33%
TB-SRGRU	0.7637	<i>0.47%</i>	<i>0.8642</i>	1.56%	0.5208	<i>0.93%</i>	0.5312	1.01%

Table 2. Comparison performance on variety of SRGRU models on four datasets in terms of the BLEU and the error rate ERR(%) scores. The results were averaged over 5 randomly initialized networks for each proposed model. ^areported in [22].

Model	Restaurant		Hotel		Laptop		TV	
	BLEU	ERR	BLEU	ERR	BLEU	ERR	BLEU	ERR
SCLSTM ^a	0.7211	0.62%	0.8020	0.78%	-	-	-	-
+deep ^a	0.7310	0.46%	0.8320	0.41%	-	-	-	-
GRU-Base	0.7208	1.55%	0.8426	1.97%	0.5158	1.94%	0.5244	2.11%
SRGRU-Base	0.7526	1.33%	0.8622	1.12%	0.5165	1.79%	0.5311	1.56%
SRGRU-Context	0.7614	0.99%	0.8677	1.75%	0.5182	1.41%	0.5312	1.37%
TB-SRGRU	0.7608	0.88%	0.8584	1.63%	0.5188	1.35%	0.5316	1.27%

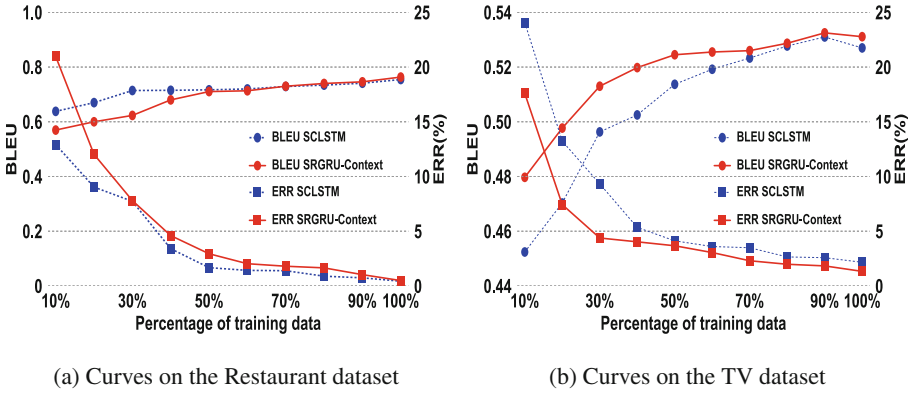


Fig. 2. Comparison of two generators SRGRU-Context and SCLSTM which are trained with different proportion of training data.

the higher slot error rate ERR. However, using the backward SRGRU reranker can improve the results in both evaluation metrics. This reranker provides benefit to the generator on producing higher-quality utterances.

Figure 2 compares two generators trained with different proportion of data evaluated on two metrics. As can be seen in Fig. 2a, the SCLSTM model achieves better results than SRGRU-Context model on both of BLEU and ERR scores since a small amount of training data was provided. However, the SRGRU-Context obtains the higher BLEU score and slightly higher ERR score as more training data was fed. On the other hand, in a more diverse dataset TV, the SRGRU-Context model consistently outperforms the SCLSTM on both evaluation metrics no matter how much training data is (Fig. 2b). This is mainly due to the ability of refinement gate which feeds to the GRU model a new input \mathbf{x}_t

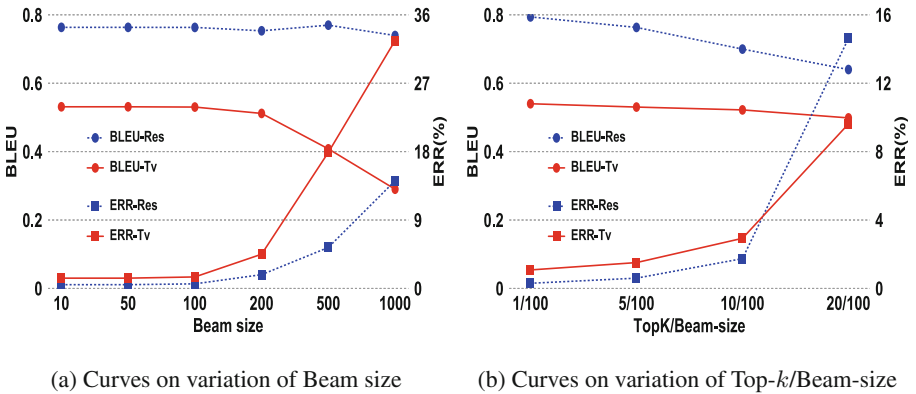


Fig. 3. RGRU-Context generator was trained with different Beam size (a) and Top-k best results (b) and evaluated on Restaurant and TV datasets.

Table 3. Comparison of top responses generated for some input dialogue acts between different models. Errors are marked in color (**missing**, **misplaced**, **repeated**, **grammar information**). ^a and ^bdenotes the baselines and the proposed models, respectively.

Model	Generated responses
Input DA	<i>inform_count(type = restaurant; count = 2; food = Basque; kidsallowed = no; pricerange = moderate)</i>
Reference	<i>There are 2 restaurants where no children are allowed in the moderate price range and serving Basque food</i>
ENCDEC ^a	There are 2 restaurants that are moderately priced and do not allow kids. [Basque]
HLSTM ^a	There are 2 Basque restaurants that are moderately priced and does not allow kids
SCLSTM ^a	There are 2 Basque restaurants that are moderate and does not allow children
SRGRU-C ^b	There are 2 moderate restaurants that serve Basque food and does not allow children
TB-SRGRU ^b	There are 2 moderately priced Basque restaurants that do not allow kids
Input DA	<i>inform(name = the Carriage Inn; hasinternet = yes; dogsallowed = no)</i>
Reference	<i>The Carriage Inn is a nice place, it has internet and where no dogs are allowed</i>
ENCDEC ^a	The Carriage Inn does not allow dogs. [has internet]
HLSTM ^a	The Carriage Inn does not allow dogs, do not allow dogs [do not allow dogs]
SCLSTM ^a	The Carriage Inn does not allow dogs and has internet
SRGRU-C ^b	The Carriage Inn has internet and does not allow dogs
TB-SRGRU ^b	The Carriage Inn does not allow dogs and it has internet
Input DA	<i>compare(name = Triton 52; ecorating = A+; family = L7; name = Hades 76; ecorating = C; family = L9)</i>
Reference	<i>Compared to Triton 52 which is in the A+ eco rating and is in the L7 product family, Hades 76 is in the C eco rating and is in the L9 product family. Which one do you prefer?</i>
ENCDEC ^a	The Triton 52 has an A+ eco rating, the Hades 76 in the L7 product family and has an C eco rating. [L7, L9]
HLSTM ^a	The Triton 52 is in the L7 product family with an A+ eco rating, while the Hades 76 has a C eco rating, which do you prefer? [L9]
SCLSTM ^a	The Triton 52 has an A+ eco rating, the Hades 76 is in the L7 family and has a eco rating of C. [L7, L9]
SRGRU-C ^b	The Triton 52 is in the L7 product family and an A+ eco rating, the Hades 76 is in the L9 family and has an C eco rating
TB-SRGRU ^b	The Triton 52 has an A+ eco rating, in the L7 product family, the Hades 76 has a C eco rating and is in the L9 product family

conveying useful information filtered from the original input and the gating mechanism. Moreover, this gate also keeps the pattern of the generated utterance during generation. As a result, it can have a better realization of unseen slot-value pairs.

Figure 3a shows an effect of beam size on the SRGRU-Context model evaluated on Restaurant and TV datasets. As can be seen that, the model performs worse in terms of degrading the BLEU score and upgrading the slot error rate ERR when the beam size increases. The model seems to perform best with beam size less than 100. Figure 3b presents an effect of top- k best results in which we fixed the beam size at 100 and top- k best results varied as $k = 1, 5, 10$ and 20. In each case, the BLEU and the error rate ERR scores were computed on Restaurant and TV datasets. The results are consistent with Fig. 3a in which the BLEU and ERR scores get worse as more top- k best utterances were chosen.

Table 3 shows comparison of top responses generated by different models for given DAs. Firstly, both models SCLSTM and SRGRU-Context seem to produce the same kind of error, for example, *grammar* mistakes or *missing* information, partly because of using the same idea about gating mechanism. However, TB-SRGRU, with tying the refinement gate, has ability to fix this problem and produce the correct utterances (row 1 of Table 3). Secondly, as noted earlier, one problem of the previous methods is the ability to handle the *binary* slot and slots that take *don't_care* value. Both SCLSTM and the proposed models are able to handle this problem (row 2 of Table 3). Finally, the TV dataset is more diverse and much harder than the others because the order of slot-value pairs should be considered during generation. For example, to generate a comparison sentence of 2 items *Triton 52* and *Hades 76* for given dialogue act *compare(name = Triton 52; ecorating = A+; family = L7; name = Hades 76; ecorating = C; family = L9)*, the generator should consider that *A+* and *L7* values belong to the former item while *C* and *L9* values to the latter. The HLSTM tends to make the *repeated* information error while the ENCDEC and SCLSTM seem to *misplace* the slot value during generation. Take *L7* value, for instance, which should be generated follow the *Triton 52* instead of *Hades 76* as in row 3 of Table 3. We found that both proposed models SRGRU-Context and TB-SRGRU can deal with this problem to generate appropriate utterances.

6 Conclusion and Future Work

We propose a gating mechanism GRU-based generator, in which we introduced a refinement gate to semantically refine the original input tokens. The refined inputs conveying meaningful information are then fed into the GRU cell. The proposed models can learn from the unaligned data to produce natural language responses conditioned on the given DA. We extensively evaluated our model on four NLG datasets and compared against the previous generators. The results show that the proposed models obtain better performance than the existing generators on all of four NLG domains in terms of the BLEU and ERR metrics. In the future, we plan to further investigate the gating mechanism to multi-domain NLG since the refinement gate shows its ability to handle the unseen slot-value pairs.

Acknowledgment. This work was supported by the JSPS KAKENHI Grant number JP15K16048.

References

1. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., et al.: Tensorflow: large-scale machine learning on heterogeneous distributed systems. arXiv preprint [arXiv:1603.04467](https://arxiv.org/abs/1603.04467) (2016)
2. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint [arXiv:1409.0473](https://arxiv.org/abs/1409.0473) (2014)
3. Cheyer, A., Guzzoni, D.: Method and apparatus for building an intelligent automated assistant, US Patent 8,677,377, 18 March 2014
4. Dušek, O., Jurčiček, F.: Sequence-to-sequence generation for spoken dialogue via deep syntax trees and strings. arXiv preprint [arXiv:1606.05491](https://arxiv.org/abs/1606.05491) (2016)
5. Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural Comput.* **9**(8), 1735–1780 (1997)
6. Karpathy, A., Fei-Fei, L.: Deep visual-semantic alignments for generating image descriptions. In: *Proceedings of CVPR*, pp. 3128–3137 (2015)
7. Lowe, R., Pow, N., Serban, I., Charlin, L., Pineau, J.: Incorporating unstructured textual knowledge sources into neural dialogue systems. In: *NIPS Workshop MLNLU* (2015)
8. Mairesse, F., Young, S.: Stochastic language generation in dialogue using factored language models. *Comput. Linguist.* **40**(4), 763–799 (2014)
9. Mikolov, T.: Recurrent neural network based language model (2010)
10. Oh, A.H., Rudnicky, A.I.: Stochastic language generation for spoken dialogue systems. In: *Proceedings of NAACL. ACL* (2000)
11. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: BLEU: a method for automatic evaluation of machine translation. In: *Proceedings of ACL*, pp. 311–318. *ACL* (2002)
12. Pennington, J., Socher, R., Manning, C.D.: Glove: global vectors for word representation. In: *EMNLP*, vol. 14 (2014)
13. Ratnaparkhi, A.: Trainable methods for surface natural language generation. In: *Proceedings of NAACL. ACL* (2000)
14. Rieser, V., Lemon, O., Liu, X.: Optimising information presentation for spoken dialogue systems. In: *Proceedings of ACL*, pp. 1009-1018. *ACL* (2010)
15. Stent, A., Prasad, R., Walker, M.: Trainable sentence planning for complex information presentation in spoken dialog systems. In: *Proceedings of ACL*, p. 79. *ACL* (2004)
16. Tran, V.K., Nguyen, L.M.: Natural language generation for spoken dialogue system using RNN encoder-decoder networks. In: *CoNLL 2017* (2017)
17. Vinyals, O., Toshev, A., Bengio, S., Erhan, D.: Show and tell: a neural image caption generator. In: *CVPR* (2015)
18. Wang, B., Liu, K., Zhao, J.: Inner attention based recurrent neural networks for answer selection (2016)
19. Wen, T.H., Gašić, M., Kim, D., Mrkšić, N., Su, P.H., Vandyke, D., Young, S.: Stochastic language generation in dialogue using recurrent neural networks with convolutional sentence reranking. In: *Proceedings of SIGDIAL. ACL* (2015)
20. Wen, T.H., Gasic, M., Mrksic, N., Rojas-Barahona, L.M., Su, P.H., Vandyke, D., Young, S.: Multi-domain neural network language generation for spoken dialogue systems. arXiv preprint [arXiv:1603.01232](https://arxiv.org/abs/1603.01232) (2016)

21. Wen, T.H., Gašić, M., Mrkšić, N., Rojas-Barahona, L.M., Su, P.H., Vandyke, D., Young, S.: Toward multi-domain language generation using recurrent neural networks (2016)
22. Wen, T.H., Gašić, M., Mrkšić, N., Su, P.H., Vandyke, D., Young, S.: Semantically conditioned LSTM-based natural language generation for spoken dialogue systems. In: Proceedings of EMNLP. ACL (2015)
23. Wen, T.H., Vandyke, D., Mrksic, N., Gasic, M., Rojas-Barahona, L.M., Su, P.H., Ultes, S., Young, S.: A network-based end-to-end trainable task-oriented dialogue system. arXiv preprint [arXiv:1604.04562](https://arxiv.org/abs/1604.04562) (2016)
24. Zhang, X., Lapata, M.: Chinese poetry generation with recurrent neural networks. In: EMNLP, pp. 670–680 (2014)



Discovering Representative Space for Relational Similarity Measurement

Huda Hakami^(✉) , Angrosh Mandya, and Danushka Bollegala

Computer Science Department, University of Liverpool, Liverpool, UK
{hshhakam, angrosh.mandya, danushka.bollegala}@liv.ac.uk

Abstract. Relational similarity measures the correspondence of the semantic relations that exist between the two words in word pairs. Accurately measuring relational similarity is important for various natural language processing tasks such as, relational search, noun-modifier classification, and analogy detection. Despite this need, the features that accurately express the relational similarity between two word pairs remain largely unknown. So far, methods have been proposed based on linguistic intuitions such as the functional space proposed by Turney [1], which consists purely of verbs. In contrast, we propose a data-driven approach for discovering feature spaces for relational similarity measurement. Specifically, we use a linear-SVM classifier to select features using training instances, where two pairs of words are labeled as analogous or non-analogous. We evaluate the discovered feature space by measuring the relational similarity for relational classification task in which we aim to classify a given word-pair to a specific relation from a predefined set of relations. Linear classifier for ranking the best feature for relational space has been compared with different methods namely, Kullback Leibler divergence (KL), Pointwise Mutual Information (PMI). Experimental results show that our proposed classification method accurately discovers a discriminative features for measuring relational similarity. Furthermore, experiments show that the proposed method requires small number of relational features while still maintaining reasonable relational similarity accuracy.

Keywords: Relational similarity · Feature selection
Proportional analogy detection

1 Introduction

Identifying the semantic relations that exist between two words (or entities) is one of the fundamental steps in many natural language processing (NLP) tasks. For example, to detect word analogies between pairs of words [2–4] such as (*water, pipe*) and (*electricity, wire*), we must first identify the relations that exist between the two words in each word pair (in this case *flows in*). In relational information retrieval [5], given a query *x is to y as z is to?* we would like to retrieve entities that have a semantic relationship with *z* similar to that between

x and y . For example, given the relational search query *Bill Gates* is to *Microsoft* as *Steve Jobs* is to?, a relational search engine is expected to return the result *Apple Inc.*

Despite the wide applications of relations in NLP systems, it remains a challenging task for humans to come up with representative features for identifying the semantic relation between two given words. In our previous example, the relationship between *Bill Gates* and *Microsoft* can be complex as Bill Gates is both a founder, a lead developer in many products, and a former CEO of the Microsoft. In order for a human to suggest representative features for identifying a relationship given only via an entity-pair instance, he/she must not only be familiar with the individual entities, but also know the different relations that would exist between those entities. Therefore, more automated methods for representing relations using descriptive features are necessary.

A popular strategy for representing the relation between two words is to extract lexical or syntactic patterns from the co-occurrence contexts of those words [6]. The extracted lexical patterns can then be used to measure the relational similarity between two word-pairs using a similarity measure defined over the distributions of patterns. Although surface patterns have been used successfully to represent the semantic relations between two words, it suffers from the data sparseness. The co-occurrences of two words with a specific pattern can be sparse even in a large corpus, requiring some form of a dimensionality reduction in practice [7]. It is also computationally expensive method because we must consider co-occurrences between surface patterns and all pairs of words. The number of all pairwise combinations between words grows quadratically with the number of words, and we require a continuously increasing set of surface patterns to cover the relations that exist between the two words in each of those word-pairs.

To overcome the above mentioned issues in the holistic approach, Turney [1, 8] proposed the *Dual Space* approach, where the relations between two words is *composed* using features related to individual words. Specifically, he used *nouns* and *verbs* as features for describing respectively the *domain* and *function* spaces. The proposal to use verbs as a proxy for the functional attributes of words that are likely to contribute towards semantic relations is based on linguistic intuition. Although this intuition is justified by the experimental results, the question *can we learn descriptors of semantic relations from labeled data?* remains unanswered.

We address this question by proposing a method for ranking lexical descriptors for representing semantic relations that exist between two words. Given a set of word-pairs for a particular relation type, we model the problem of extracting descriptive features as a linear classification problem. Specifically, we train a linear-SVM to discriminate between positive (analogous) and randomly generated pseudo-negative (non-analogous) word-pairs using features associated with individual words. The weights learnt by the classifier for the features can then be used as a ranking-score for selecting most representative features for a particular semantic relation. Experimental results on a benchmark dataset for relation

classification show that the proposed feature selection method outperforms several competitive baselines and previously proposed heuristics.

The paper is organized as follows: in Sect. 2 we discuss some related work of feature selection in NLP. The methodology adopted in this work is presented in Sects. 3 and 4. The dataset applied in this research with the experimental results are discussed in Sect. 5. Finally, we conclude the paper and discuss some possible future works.

2 Related Work

Identifying appropriate feature space for NLP tasks is a problem that have been studied widely in the literature. The most popular and effective method is based on matrix factorisation such as Non-Negative Matrix Factorization (NMF), Principle Component Analysis (PCA) and Singular Value Decomposition (SVD). Basically, those methods aim to transform the high-dimensional distributional representations to low-dimensional latent space. For word-level representation, Latent Semantic Analysis (LSA) is a method relying on SVD to represent a word in a vector space using only top, i.e. 300 or more, dimensions to capture the meaning of words in the low-dimensional latent space [9]. For word pairs representation, Latent Relational Analysis (LRA) is a method proposed by Turney [10] for measuring the similarity in the semantic relations between two pairs of words. In LRA, SVD has been applied to pair-pattern matrix to represent a latent feature space. Although LRA achieve satisfied result for answering the 374 SAT questions (56.1%), it is complex process to factorize a huge matrix and thus it is time-consuming method (requires 9 days to run).

On the other hand, many feature selection methods have been proposed in the literature. Selecting important features using classification approach has been used for different NLP tasks such as sentiment analysis [11] and text classification [12, 13]. Given a number of examples for specific task, linear classifier ables to recover the features that are relevant to separate the examples into classes. For example, in text classification a documents are represented by words in the vocabulary which suffer from the curse of dimensionality. A linear classifier generates coefficients of the features in the space which are used to rank the most informative words that helps in separating documents into categories.

For sentence-level similarity, Ji and Eisenstein [14] apply data-driven approach for weighting the features for paraphrase classification task. Based on supervised (labeled) dataset, they propose new weighting metric for features in order to distinguish the deterministic features for sentence semantics. The weighting metric uses KL Divergence to weight the distributional features in the co-occurrence matrix for sentences before decomposing process. They report significant improvement on sentence similarity in comparison with other works.

Another approach to select a subset of informative feature is using mutual information based methodology. PMI statistical weighting method has been applied for feature selection for document categorisation [15, 16]. It calculates the amount of information that a feature includes about a specific categories.

Xu et al. [15] show that MI is not efficient approach to select relevant feature for text classification compared with other known approach such as Document Frequency (DF) and Information Gain (IG).

While there are efforts spent for feature selection for many NLP tasks, only few attentions have been directed to relational similarity between two pairs of words. Turney [1] heuristically identify a space for semantic relations called function space which consist of verb patterns. For example, for an analogy (*word, language*), (*note, music*), *word* and *note* share the same function, e.g. the function of building units (*vocabularies*). Similarity, *language* and *music* share the same function, the function of *communications*. To the best of our knowledge, there is no work yet on feature selection data-driven methods for relational similarity task. Consequently, this paper contribute to handle that issue.

3 Relational Similarity in Feature Space

Let us consider a feature x in some feature space \mathcal{S} . We do not impose any constraints on the type of features here, and the proposed method can handle any type of features that can be used to represent a word such as other words that co-occur with a target word in the corpus (lexical features), or their syntactic categories such as part-of-speech (POS) (syntactic features). The feature space \mathcal{S} is defined as the set containing all features we extract for all target words. We represent the salience of x in \mathcal{S} by the discriminative weight $w(x, \mathcal{S}) \in \mathbb{R}$. For example, if x is a representative feature of \mathcal{S} , then it will have a high $w(x, \mathcal{S})$. The concept of a discriminative weight can be seen as a feature selection method. If a particular feature is not a good representative of the space, then it will receive a small (ideally zero) weight, thereby effectively pruning out the feature from the space.

Given the above setting, the task of discovering relational feature spaces can be modelled as a problem of computing the discriminative weights for features. We use $\phi(A)$ to denote the set of non-zero features that co-occur with the word A . The salience $f(A, x, \mathcal{S})$ of x as a feature of A in \mathcal{S} is defined as:

$$f(A, x, \mathcal{S}) = h(A, x) \times w(x, \mathcal{S}) \quad (1)$$

Here, $h(A, x) \geq 0$ is the strength of association between A and x , and can be computed using any non-negative feature co-occurrence measure. In our experiments we use positive pointwise mutual information (PPMI) computed using corpus counts as $h(A, x)$.

(1) is analogous to the tf-idf score used in information retrieval in the sense that $h(A, x)$ corresponds to the term-frequency (tf) (i.e. how significant is the presence of x as a feature in A), and $w(x, \mathcal{S})$ corresponds to the document-frequency (df) (i.e. what is the importance of x as a feature in the space \mathcal{S}). The similarity, $\text{sim}_{\mathcal{S}}(A, C)$ between two words A and C in \mathcal{S} can then be defined as in (2) which is the sum of pointwise products over the intersection of the feature sets $\phi(A)$ and $\phi(C)$.

$$\text{sim}_{\mathcal{S}}(A, C) = \sum_{x \in \phi(A) \cap \phi(C)} f(A, x, \mathcal{S}) f(C, x, \mathcal{S}) \quad (2)$$

Moreover, by substituting (1) in (2) we get:

$$\text{sim}_{\mathcal{S}}(A, C) = \sum_{x \in \phi(A) \cap \phi(C)} h(A, x)h(C, x)w(x, \mathcal{S})^2 \quad (3)$$

Following the proposal by [1], we can then compute the relational similarity, $\text{sim}_{\text{rel}}((A, B), (C, D))$, between two word-pairs (A, B) and (C, D) as the geometric mean of their functional similarities:

$$\text{sim}_{\text{rel}}((A, B), (C, D)) = \sqrt{\text{sim}_{\mathcal{S}}(A, C) \times \text{sim}_{\mathcal{S}}(B, D)} \quad (4)$$

4 Learning Features Weights

The relational similarity measure described in Sect. 3 depends on the feature space \mathcal{S} via the discriminative weights $w(x, \mathcal{S})$ assigned to each feature x . Therefore, our goal of discovering a representative feature space from data can be seen as a problem of learning $w(x, \mathcal{S})$. We propose a supervised classification-based approach for computing discriminative weights using labeled dataset.

Let us denote a labeled dataset consists of word-pairs (A, B) and (C, D) annotated for $l = 1$ (i.e. the two word pairs are analogous) or $l = 0$ (otherwise). Here, $l \in \{0, 1\}$ denotes the class label. From (12) and (3), we see that for two analogous word-pairs, (A, B) and (C, D) , their relational similarity increases if the two products $h(A, x)h(C, x)$ and $h(B, x)h(D, x)$ increase. Following this observation, we define a feature x to appear in an instance word-pairs (A, B) and (C, D) iff:

$$(x \in \phi(A) \cap \phi(C)) \vee (x \in \phi(B) \cap \phi(D)) \quad (5)$$

4.1 Linear Classifier Method for Relational Feature Ranking

For the proposed classification-based approach, each positive instance word pairs $((A, B), (C, D))$ or negative word-pairs $((A', B'), (C', D'))$ have a corresponding feature vector in \mathcal{S} , such that the entry for x in the $(A, B), (C, D)$ positive instance is defined as follows:

$$g(((A, B), (C, D)), x) = \mathcal{I}[x \in \phi(A) \cap \phi(C)] + \mathcal{I}[x \in \phi(B) \cap \phi(D)] \quad (6)$$

Here, $g(((A, B), (C, D)), x)$ denotes the value of feature x in the feature vector representing the instance $((A, B), (C, D))$, and \mathcal{I} is the indicator function which return 1 if the expression evaluated is true, or 0 otherwise. Likewise for a negative instance. We train a linear-SVM binary classifier to learn a weight for each feature in the feature space. $w(x, \mathcal{S})$ can be interpreted as the confidence of the feature as an indicator of the strength of analogy (relational similarity) between (A, B) and (C, D) . The absolute value of a weight of a feature can be considered as a measure of the importance of that feature when discriminating the two classes in a binary linear classifier. Therefore, we rank the features in the space according to the absolute value of the weights $|w(x, \mathcal{S})|$. Only linearised kernel classifier explicitly associates weights to individual features. Therefore,

this approach is restricted to linear kernel. In the case of non-linear kernels such as polynomial kernels that can be expanded prior to learning to all feature combinations considered in the kernel computation, we can still apply this technique to identify salient feature combinations. However, we limit the discussion in this paper to finding relational feature spaces consisting of individual features and defer the study of salient feature combinations for relational similarity measurement to future work.

The proposed method is compared against baseline methods namely: KL and PMI in addition to random selection and heuristic verb space. KL and PMI methods also require labelled data as in the proposed classification-based approach.

4.2 KL Divergence-Based Ranking Approach

We consider KL divergence-based weighting approach proposed by [14] to compute $w(x, \mathcal{S})$ for relational similarity measurement. For this purpose, we will consider the two distributions for each feature x in \mathcal{S} -space namely, $p(x)$ and $q(x)$ where $p(x)$ is computed for analogous $((A, B), (C, D))$, while $q(x)$ is taken over the unrelated pairs of words $((A', B'), (C', D'))$. $p(x) = P(x \in \phi(A) | x \in \phi(C), l = 1 \text{ or } x \in \phi(B) | x \in \phi(D), l = 1)$. Similarly, $q(x) = P(x \in \phi(A') | x \in \phi(C'), l = 0 \text{ or } x \in \phi(B') | x \in \phi(D'), l = 0)$.

Specifically, we compute the probability $p(x)$ of a feature x being an indicator of the analogous class as follows:

$$\frac{1}{Z_p(x)} \sum_{(A,B),(C,D) \in \mathcal{N}_+} g(((A, B), (C, D)), x) \quad (7)$$

Here, \mathcal{N}_+ is the set of positive word-pairs, and the normalisation coefficient $Z_p(x)$ satisfies, $\sum_{x \in \mathcal{S}} p(x) = 1$. Likewise, we can compute $q(x)$, the probability of a feature x being an indicator of the negative (relationally dissimilar) class using the features occurrences in negative instances $((A', B'), (C', D'))$ as follows:

$$\frac{1}{Z_q(x)} \sum_{(A',B'),(C',D') \in \mathcal{N}_-} g(((A', B'), (C', D')), x) \quad (8)$$

Here, \mathcal{N}_- is the set of negative word-pairs, and the normalization coefficient $Z_q(x)$ satisfies, $\sum_x q(x) = 1$. Having computed $p(x)$ and $q(x)$, we then compute $w(x, \mathcal{S})$ as the KL divergence between the two distributions as,

$$w(x, \mathcal{S}) = p(x) \log \left(\frac{p(x)}{q(x)} \right). \quad (9)$$

4.3 PMI Ranking Approach

PMI is used to weight a feature x such that:

$$w(x, \mathcal{S}) = \text{PMI}(x, \mathcal{N}_+) - \text{PMI}(x, \mathcal{N}_-) \quad (10)$$

Where $\text{PMI}(x, \mathcal{N}_+)$ measures the association between a feature x with analogues word-pairs, whereas $\text{PMI}(x, \mathcal{N}_-)$ indicates the co-occurrence of a feature with relationally dissimilar pairs. PMI has been computed as follows:

$$\text{PMI}(x, \mathcal{N}_+) = \log \left(\frac{h(x, \mathcal{N}_+)}{h(x, \mathcal{N})|\mathcal{N}_+|} |\mathcal{N}| \right) \quad (11)$$

$$\mathcal{N} = \mathcal{N}_+ \cup \mathcal{N}_-$$

Here \mathcal{N} is the union set of the positive and negative word-pairs and $h(x, \mathcal{N}_+)$ is summed for all analogous pairs:

$$\sum_{(A,B),(C,D) \in \mathcal{N}_+} g(((A, B), (C, D)), x)$$

Similarly, $h(x, \mathcal{N}_-)$ is calculated considering negative instances in the dataset.

We rank the features according to the absolute values of their weights by each of the methods described to define the representative space to measure the relational similarity. The relational similarity between two given word pairs is computed as follows after reducing the word representations to the top ranked feature space:

$$\text{sim}_{\text{rel}}((A, B), (C, D)) = \sqrt{\text{sim}(A, C) \times \text{sim}(B, D)} \quad (12)$$

Cosine similarity between two vectors is defined as follows:

$$\text{sim}(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x}^\top \mathbf{y}}{\|\mathbf{x}\| \|\mathbf{y}\|} \quad (13)$$

We experimented using both unnormalised word embeddings as well as ℓ_2 normalised word representations. We found that ℓ_2 normalised word representations perform better than the unnormalised version in most configurations. Consequently, we report results obtained only with the ℓ_2 normalised word representations in the remainder of the paper.

5 Experimental Design

5.1 Dataset

The above mentioned feature selection methods require a labelled dataset of word-pairs for a particular relation type. To generate such a dataset we use the following procedure. We used the DIFFVECS dataset proposed by Vylomova et al. [17] that consists of triples $\langle w_1, w_2, r \rangle$, where word w_1 and w_2 are connected by a relation r ¹. This dataset consists of 15 relation types, we include the relation types for which we have efficient number of pairs to generate the dataset. Consequently, 7 semantic relation types and their sub categories have been considered in this study as presented in Table 1.

¹ <https://github.com/ivri/DiffVec>.

Table 1. Statistic of the dataset used in this study.

Relation type	Sub-relations	Example of positive instance	No. of Pos instances	No. of testing pairs
Hypernym	–	(<i>colour : green</i>) (<i>tool : knife</i>)	1,100	57
Meronym	–	(<i>dishwasher : door</i>) (<i>tiger : mouth</i>)	1,100	57
Event (objects action)	–	(<i>arrive : train</i>) (<i>fix : oven</i>)	1,100	57
Cause-Purpose	Enabling-Agent: Object, Cause: Effect, Agent: Goal, Prevention	(<i>eating : fullness</i>) (<i>illness : discomfort</i>)	1,149	56
Space-Time	Item: Location, Location: Process	(<i>library : reading</i>) (<i>park : playing</i>)	1,435	56
Reference	Plan, Sign: Significant Expression, Representation	(<i>red : stop</i>) (<i>warning : trouble</i>)	1,047	54
Attribute	Object: TypicalAc- tion(noun.verb) ObjectState (noun.noun)	(<i>musician : sing</i>) (<i>tree : grow</i>)	256	30
Total	–	–	7,187	367

For each relation, we exclude some pairs of words for testing the methods, in total we have 367 testing pairs distributed among the relations. We generate positive training instances by pairing word-pairs that have same relation types (considering sub-relations), resulting in 7,187 positive instances from this procedure. Next, we randomly pair a word-pair from a relation r with a word-pair from a relation r' such that $r \neq r'$ to create a pseudo-negative training dataset that has approximately an equal number of instances as that in the positive training dataset (i.e., 7,000).

5.2 Evaluation Measures

During evaluation, we consider the problem of classifying a given pair of words (w_1, w_2) to a specific relation r in a predefined set of relations \mathcal{R} according to the relation that exists between w_1 and w_2 . We measure the relational similarity between a given pair and all the remaining pairs in the testing data. Then, we perform 1-NN relation classification such that if the 1-NN has the same relation label as the target pair, then we consider it to be a correct match. Macro-averaged classification accuracy is used as the evaluation measure. We use the

PPMI matrix from Turney [18], which contains PPMI values between a word and unigrams from the left and right contexts of that word in a corpus². The total number of features extracted ($|\mathcal{S}|$) is 139,246.

5.3 Results

For a classification method, we train linear SVM using scikit-learn library³. We use 5 folds cross-validation to find the optimal value of penalty parameter C of the error term. Following Turney [1], we used verbs as \mathcal{S} to evaluate the performance of the functional space for measuring relational similarity. We used the NLTK POS tagger⁴ for identifying verbs in the feature space. The verb space identified by the POS tagger contains 12k verbs.

In Table 2, we compare the feature weighting methods discussed in Sect. 4 for different semantic relation types used in the evaluated dataset (illustrated in Table 1). The accuracies for SVM-based, KL, PMI and random ranking methods are reported for the top 1k features. For verb-space, the results indicate the performance of the 12k verbs in the feature space. Classification approach of weighting features and verb-space perform equally for hypernym relation. For meronym, event and attribute relation types the proposed linear-SVM outperforms other methods of feature ranking. KL divergence-based method shows its ability to perform well compared with other methods for cause-purpose and space-time relations. Among different relation types compared in Table 2, classification-based weighting method reports the highest macro-average accuracy compared with other baselines. The fact that the proposed method could improve the performance for many relations of relational classification task empirically justifies our proposal for a data-driven approach for feature selection for relational similarity measurement.

Table 2. Accuracy per relation type for the top 1000 ranked features.

Relation	Classifier	KL	PMI	Verb-space	Random
Hypernym	73.68	71.93	56.14	73.68	54.39
Meronym	70.18	68.42	45.61	61.4	56.14
Event	78.95	73.68	29.82	66.67	54.39
Attribute	33.33	13.33	30.00	23.33	10.00
Cause-Purpose	41.07	44.64	28.57	37.50	21.43
Space-Time	58.93	64.29	33.93	62.5	46.43
Reference	57.41	59.26	42.59	64.81	33.33
Macro-average	59.08	56.51	38.10	55.7	39.44

² The corpus was collected by Charles Clarke at the University of Waterloo.

³ <http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html>.

⁴ <http://www.nltk.org>.

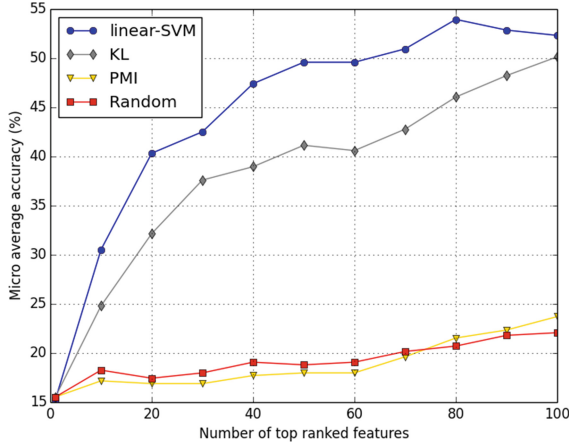


Fig. 1. Cumulative evaluation of feature weighting methods.

We evaluate which of the ranking methods ranks the relational features at the top of the weighted feature list. Figure 1 shows the micro-average accuracies of the top-ranked features selected by the different methods, verb-space is not included in this comparison as it is not a ranking method for feature selection. We start by evaluating the top ranked feature, subsequently adding 10 more features at a time. The random baseline randomly selects a subset of features from \mathcal{S} . As shown in the Fig. 1, the top-weighted features using the proposed linear SVM-based approach outperforms all other methods for relational similarity measurement. The proposed method statistically significantly outperforms (according to McNemar test with $p < 0.05$) all other methods for ranking the most informative features in the top ranked feature list. This indicates that the effective features for measuring relational similarity are indeed ranked at the top by the proposed method. In addition, our results show that it is possible to maintain a relational classification accuracy while using only small subset of the features (top 100 features). KL divergence-based ranking method follow classification approach for ranking the best features for relational similarity. However, PMI method performs badly as it gives accuracies comparable with the random feature selection method. PMI is known to give higher values to rare features thereby preferring rare features. We believe this might be an issue when selecting features for representing word-pairs.

6 Conclusion

We proposed the first-ever method for discovering a discriminative feature space for measuring relational similarity from data. The relational classification results show that using labeled data to train a linear classifier for feature selection can improve the feature space for relational similarity measurement. The proposed method outperforms KL and PMI methods for discovering relational

feature space. Using PMI to discover relational features has been demonstrated to have relatively poor performance, a finding which is consistent with previous work for text classification task [15]. In addition, classification-based weighting method reports better performance for many relation types compared with the functional verb space. Future researches can be carried out to improve the feature space for relational similarity task by incorporating verb space with the data-driven discovered features.

References

1. Turney, P.D.: Domain and function: a dual-space model of semantic relations and compositions. *J. Artif. Intell. Res.* **44**, 533–585 (2012)
2. Bollegala, D., Matsuo, Y., Ishizuka, M.: A relational model of semantic similarity between words using automatically extracted lexical pattern clusters from the web. In: *Proceedings of the Empirical Methods in Natural Language Processing*, pp. 803–812 (2009)
3. Turney, P.D.: A uniform approach to analogies, synonyms, antonyms, and associations. In: *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pp. 905–912 (2008)
4. Nakov, P., Kozareva, Z.: Combining relational and attributional similarity for semantic relation classification. In: *Proceedings of the Recent Advances in Natural Language Processing*, pp. 323–330 (2011)
5. Duc, N.T., Bollegala, D., Ishizuka, M.: Using relational similarity between word pairs for latent relational search on the web. In: *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, pp. 196–199 (2010)
6. Riedel, S., Yao, L., McCallum, A., Marlin, B.M.: Relation extraction with matrix factorization and universal schemas. In: *Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 74–84 (2013)
7. Turney, P.D.: Similarity of semantic relations. *Comput. Linguist.* **32**(3), 379–416 (2006)
8. Turney, P.D.: Distributional semantics beyond words: supervised learning of analogy and paraphrase. *Trans. Assoc. Computat. Linguist.* **1**, 353–366 (2013)
9. Landauer, T.K., Foltz, P.W., Laham, D.: An introduction to latent semantic analysis. *Discourse Process.* **25**(2–3), 259–284 (1998)
10. Turney, P.D.: Measuring semantic similarity by latent relational analysis. In: *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 1136–1141 (2005). arXiv preprint [arXiv:cs/0508053](https://arxiv.org/abs/cs/0508053)
11. Tripathi, G., Naganna, S.: Feature selection and classification approach for sentiment analysis. *Mach. Learn. Appl. Int. J.* **2**(2), 1–16 (2015)
12. Brank, J., Grobelnik, M., Milic-Frayling, N., Mladenic, D.: Feature selection using support vector machines. *WIT Trans. Inf. Commun. Technol.* **28** (2002)
13. Mladenić, D., Brank, J., Grobelnik, M., Milic-Frayling, N.: Feature selection using linear classifier weights: interaction with classification models. In: *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 234–241. ACM (2004)

14. Ji, Y., Eisenstein, J.: Discriminative improvements to distributional sentence similarity. In: Proceedings of the Empirical Methods in Natural Language Processing, pp. 891–896 (2013)
15. Xu, Y., Jones, G.J., Li, J., Wang, B., Sun, C.: A study on mutual information-based feature selection for text categorization. *J. Comput. Inf. Syst.* **3**(3), 1007–1012 (2007)
16. Schneider, K.-M.: Weighted average pointwise mutual information for feature selection in text categorization. In: Jorge, A.M., Torgo, L., Brazdil, P., Camacho, R., Gama, J. (eds.) PKDD 2005. LNCS (LNAI), vol. 3721, pp. 252–263. Springer, Heidelberg (2005). https://doi.org/10.1007/11564126_27
17. Vylomova, E., Rimmel, L., Cohn, T., Baldwin, T.: Take and took, gaggle and goose, book and read: evaluating the utility of vector differences for lexical relation learning. In: Proceedings of the Association for Computational Linguistics, pp. 1671–1682 (2016)
18. Turney, P.D., Neuman, Y., Assaf, D., Cohen, Y.: Literal and metaphorical sense identification through concrete and abstract context. In: Proceedings of the Empirical Methods in Natural Language Processing, pp. 27–31 (2011)



Norms of Valence and Arousal for 2,076 Chinese 4-Character Words

Pingping Liu^{1,2,3(✉)}, Minglei Li¹, Qin Lu^{1(✉)}, and Buxin Han^{2,3}

¹ Department of Computing, The Hong Kong Polytechnic University, Hong Kong 999077, China
liupp@psych.ac.cn, csluqin@comp.polyu.edu.hk

² CAS Key Laboratory of Mental Health, Institute of Psychology, Beijing 100101, China

³ Department of Psychology, University of Chinese Academy of Sciences, Beijing, China

Abstract. This study describes an annotated dataset through psycho-linguistic annotations in controlled environment on valence and arousal for a large lexicon of 2,076 Chinese 4-character words. The purpose for the annotation is to provide affect-linked knowledge to text which can be used in affective computing using NLP techniques. Analysis to the annotated data indicates that valence and arousal fit the classical U-shaped distribution. Most importantly, the annotated results indicate that the same 2-character word that appears in different 4-character words can indeed show distinct affective meanings which implies that the affective meaning of 4-character words may not be compositional to its component words. The study on this annotated list of 4-character words not only has significance at the intersection of cognitive neuroscience and social psychology, but also has great value as a resource for affective analysis in NLP applications.

Keywords: Valence · Arousal · Chinese words · Emotion · Affective analysis

1 Introduction

Many affective linked analysis tasks such as sentiment analysis and emotion analysis require the availability of affective knowledge of words [1, 2]. Valence and Arousal (VA) form a pair of commonly used affective measures. VA has theoretical backing from psychology and cognitive science [1, 3]. It is often used in psychological experiments to measure emotion and emotional changes linked to the annotated words. Many NLP applications that need the use of affective information also use affective lexicons with annotated VA values. Many social and psychology related studies concern the ways affective words are produced and perceived, and how these words influence human cognition and behavior [4, 5]. Words associated with affective meanings can also be used to estimate the emotional values of other words or the emotions expressed by entire passages or text through computational means [6–11]. Generally speaking, the VA measures are very useful in the classification of affective experiences [1, 3]. Valence is a subjective assessment that describes how pleasant a stimulus is arousal refers to the subjective level of activation or intensity that a stimulus elicits. The studies on VA are conducted for several languages and actual datasets are acquired for different languages including English [12, 13], German [14], French [15], Spanish [16] and Chinese [17–19].

Psychology and other social science studies on the effects of affective variables for word stimuli usually need to reference well annotated lexicons, referred to as the normed lists. In English related studies, most works use the affect annotated ANEW lexicon (i.e., the Affective Norms for English Words), which provides VA values for 1,034 words [12]. In Chinese, most works on affective computing use the Chinese Affective Word System (CAWS), which provides the VA values for 1,500 2-character words [19]. Recently, Yao et al. (2016) also introduces 1,100 2-character Chinese lexicon with valence-arousal, and other semantic measures.

However, no Chinese affective lexicon has 4-character words. According to the *Chinese Lexicon*, 63.9% of Chinese words are 2-character words, and 14.2% are 4-character words [20]. In terms of frequency, 4-character words are in 13.4% of text [21]. Obviously, 4-character words are more complex than 2-character words and they can convey more complex meanings. Sometimes, the affective states of 4-character words cannot be inferred from its component words based on the principle of compositionality. For instance, the emotional meaning of the word 守株待兔 (i.e., wait for gains without pains) cannot be computed from these individual characters (守=*stay*, 株=*tree*, 待=*wait*, 兔=*hare*), or composition of the two characters (i.e., 守株 *and* 待兔 are not commonly used words in *Chinese*). Another interesting observation is that the same 2-character word may have different distinct affective meanings in different 4-character word context. For example, the word 相待 (i.e., treat each other), which is slightly positive as a 2-character word, may change its own sentiment in different context. For example, in the word 坦诚相待 (i.e., be honest to each other), it is positive whereas in the words 冷眼相待 (i.e., sarcastic to each other), it is a negative component. The affective meaning of 4-character words are worthy of separate studies rather than consider them as compositions of component words. These motivate us to produce a collection of 4-character words as a supplemental resource to the currently available 2-character affective lexicon for affective computing.

The rest of the paper is organized as follows. Section 2 presents related work. Section 3 explains the methodology for annotation. Thus, the present study was to explore the norms of valence and arousal for 4-character words. Section 4 analyses the data for its soundness. Section 5 concludes the paper.

2 Related Work

Earlier works represent affective meanings of words by discrete affective labels, such as positive, negative, happiness, sadness, anger and others [1, 3]. Another method is to represent affective meaning by the more comprehensive multi-dimensional representation models, such as the valence-arousal-dominance model (VAD) [7, 22]. There are also other multi-dimensional models based on cognition grounded theories such as the evaluation-potency-activity model (EPA) [4, 23]. Discrete models are easier to annotate, thus related resources such as polarity based lexicons are more readily available [8, 24]. Compared to discrete emotion labels or one dimensional sentiment, multi-dimensional affective representation is defined in continuous space, has the backing of more comprehensive and it can capture more fine-grained information compared to the discrete dimensional models. According to the Affective Control

Theory (ACT), each concept in an event has a transient affective meaning which is context dependent in addition to cultural, behavior and other background information [25]. Yet, the affective data under the multi-dimensional models are more difficult to acquire, and thus these seemingly good models are less used because of the lack of transferred knowledge into lexical resources. Mathematically speaking, discrete affective labels can always be mapped to certain points in a multi-dimensional affective space. Sentiment indicated by polarities can be viewed as a one dimensional affective model [3]. For example, it is equal to the valence dimension in VA or the evaluation dimension in EPA.

Due to the lack of complete annotated lexicons, there are studies to infer affective meanings of words, phrases or sentences using compositional algorithms [26, 27]. In compositional prediction models, the meaning of complex expressions is determined by the meanings of its constituent expressions and the rules used to combine them. Different composition models are proposed to infer the meanings of larger text units from the meanings of single words. For example, Mitchell and Lapata (2010) proposed to use addition and multiplication of the vector representation of component words to infer the vector representation of sentences [10]. Furthermore, Baroni and Zamparelli (2010) proposed to represent a noun as a vector and an adjective as a matrix and use matrix-vector multiplication to obtain the representation of adjective noun phrases [9]. Whether the principle of compositionality still holds in affective space is worthy of exploring. Some proposed task specific and supervised composition models include recursive neural networks by matrix multiplication on concatenated component vectors [11]. Zhao et al. (2015) proposed a tensor based composition model to learn phrase representations by vector-tensor-vector multiplication [28]. The difference is that the tensor is phrase-type sensitive, which means that learning a tensor for each type of phrases, such as adjective-noun, noun-noun phrase, verb-noun phrase. However, previous lexicon resources for affective computing focus on word level. Combining with the word level affective lexicons, our annotated 4-character lexicon can be used as gold standards to explore the principle of compositionality in affective space.

3 Methodology for Annotation

The annotation is done in a controlled psychology lab at the Institute of Psychology in Beijing¹. Data was collected from June to July in 2015.

3.1 Participants and Information Supplied

A total of 72 adults (36 female) with a mean age of 22.1 years (range: 16.0–38.4 years, $SD = 4.0$) who are students in schools or universities near the Institute of Psychology participated in the study. Their average number of years of schooling is 14.4 years ($SD = 2.6$). All participants are native Chinese speakers with normal or corrected-to-normal vision. Each participant is provided with informed consent and received a minimal

¹ CAS Key Laboratory of Mental Health, Institute of Psychology, Beijing, China.

fee for their time commitment. The annotation is conducted using rating values to valence-arousal. For quality assurance, an additional confidence score is also collected.

3.2 Materials and Division of Tasks

Three native Chinese speakers who have good linguistics knowledge are asked to select a set of 4-character words which can trigger their subjective feelings. As a result, a set of 2,290 4-character words (715 positive words, 661 neutral words and 914 negative words) are selected from the Chinese Lexicon (2003) which fit the criterion and are considered frequently used. Second, these 2,290 words are then split randomly into 5 mutually exclusive groups of 458 words each.

72 participants are asked to rate these words for valence, arousal and familiarity. Among these 72 participants, two persons completed all the 5 groups (i.e., 2,290 words), one completed 4 groups (i.e., 1,832 words), eight completed 3 groups (i.e., 1,374 words), twenty completed 2 groups (i.e., 916 words), and forty-one completed 1 group (i.e., 458 words). Each participant is allowed to annotate only one group in one day. Finally, every word is rated by at least 22 participants.

3.3 Procedure and Apparatus

Participants conducted each group of annotation individually. Participants are given a description of the task when they first arrives at the lab, and the consent form must be signed and their visual acuity be checked. The instruction for the annotation is summarized below:

- *Purpose: explores emotion and concerns how people respond to different types of words.*
- *Method: use a rating scale to evaluate how you feel for each given word (around 400 words).*
- *Ratings: ranges from 1 (extremely unhappy [calm²]) to 9 (extremely happy [excited]). Far right scale corresponds to complete happiness, pleased, satisfied, contented or hopeful [stimulated, excited, jittery, wide-awake or arousal]. For example, the word 双喜临门 (i.e., “double happiness”), is likely to trigger “completely happy” [aroused] and thus the rating should be 9. The other end of the scale is for completely unhappy, annoyed, unsatisfied, despaired or bored [relaxed, calm, dull, sleepy or unaroused]. For example, the word 凶残至极 (i.e., extremely brutal), is likely to receive rating 1 for valence-arousal rating of 9.*
- *Time: work at your own pace. Make your ratings based on your actually and immediate reaction as you read each word.*

To ascertain that participants understand the task, they are first asked to evaluate 15 practice items under the supervision of the experimenter. After passing this, they start to annotate each group of data individually. Stimuli are presented in 34-point Song font on a 17-in. monitor as black text on a grey background to prevent eye fatigue. After the

² Brackets correspond to arousal scale.

rating of each VA pair, participants need to indicate their familiarity with the word (yes or no). After every 30 words, there is a reminder for a rest interval and the participant take the rest at their pace. The annotation for each group lasts for an hour on average.

4 Processing and Analysis of the Lexicon

After the annotation is completed, the set of data for the 2,290 4-character words are further cleaned before the data is used to produce the lexicon.

4.1 Data Cleaning and Outlier Analysis

In data cleaning, 214 words are removed because they cannot be identified by more than 10% of the participants according to the familiarity. To ensure quality of data, we also examine individual participant's data to make sure that each one adequately understands the instructions and the resulting data from each one falls into a reasonably normal range. Otherwise, the data produced would be considered outliers and will be used as data points to derive the final lexicon. We define outliers as the scores either 3.0 standard deviations above or below the group average for each item. Approximately 1.2% of total 54,419 recorded responses (642) were excluded either due to missing data or outlier by the $3SD$ criterion.

4.2 Descriptive Statistics

To obtain the final lexicon, we need to first analyze the results from different participants and then use the norm from the whole collection of data points as the VA measures of the lexicon. Table 1 shows the means, standard deviations (SD), minimums (Min), and maximums (Max) for the different variables. Word complexity in the table is the sum of the number of strokes of the four characters.

To use the data as a lexicon for sentiment analysis, we need to convert the VA values into polarity. Since polarity is linked to valence only. We directly take the suggestion from prior studies [17, 19], to map negative to valence values ranging from 1 to 4 ($M = 3.22$, $SD = .46$), neutral to valence ranging from 4 to 6 ($M = 5.05$, $SD = .54$), and positive to values ranging from 6 to 9 ($M = 6.60$, $SD = .38$). According to this set of mapping, 663 words are negative (31.9%), 856 are neutral (41.2%), and 557 are positive (26.8%). To use the data for emotion analysis, VA values can either be used directly in the multi-dimensional affective space, or it can always be mapped into discrete emotional labels [29].

4.3 Reliability of the Measures

To estimate the inter-rater reliability of the ratings of the valence and arousal in the dataset, the split-half correlations are analyzed and corrected with the Spearman-Brown formula. For each word, participants are randomly divided into two subgroups of equal size. Regarding the two affective variables, the mean correlation values are $r = .94$ for

Table 1. Descriptive statistics.

Variables	Mean	SD	Min	Max
Total (N = 2076)				
Valence	4.88	1.38	1.79	7.71
Arousal	5.54	.85	3.00	7.83
Word frequency	117.5	207.8	2.00	5266
Word complexity	30.6	7.38	8.00	70.0
Negative words (N = 663)				
Valence	3.22	.46	1.79	4.00
Arousal	6.02	.63	4.44	7.83
Word frequency	114.0	259.4	2.00	5266
Word complexity	31.7	7.93	12.0	70.0
Neutral words (N = 856)				
Valence	5.05	.54	4.02	5.98
Arousal	5.08	.78	3.04	7.29
Word frequency	115.7	169.9	3.00	1403
Word complexity	30.0	7.31	8.00	56.0
Positive words (N = 557)				
Valence	6.60	.38	6.00	7.71
Arousal	5.65	.81	3.00	7.63
Word frequency	124.3	191.1	7.00	1842
Word complexity	30.2	6.65	13.0	53.0

valence and $r = .75$ for arousal ($n = 2,076$). This result shows that valence has a higher inter-rater reliability than arousal, which is consistent with prior findings [15, 17, 30].

4.4 Relationship Among Word Variables

Based on the statistics given in Table 1, we further explore the relationships between valence and arousal as well as their relationship with word frequency.

Relationship between valence and arousal

Previous studies in different languages have commonly reported that valence and arousal are highly related [17]. The relation of these two variables in our dataset is analyzed using regression models. Results show that the quadratic model [$R^2 = 0.336$; $F(2, 2073) = 527.0$, $p < .001$] outperforms the simpler linear model [$R^2 = 0.051$; $F(1, 2074) = 111.9$, $p < .001$]. Figure 1 shows the result of the quadratic regression with the mean valence and its square as independent variables and the mean arousal as a dependent variable. These results show that highly positive and negative 4-character words are rated as being the most arousing stimuli whereas words with low positive and negative ratings are perceived as being the least arousing. This reliable tendency is consistent with previous studies in that the relationship between valence and arousal is observed indeed form a U-shaped distribution in different languages [14, 15, 17].

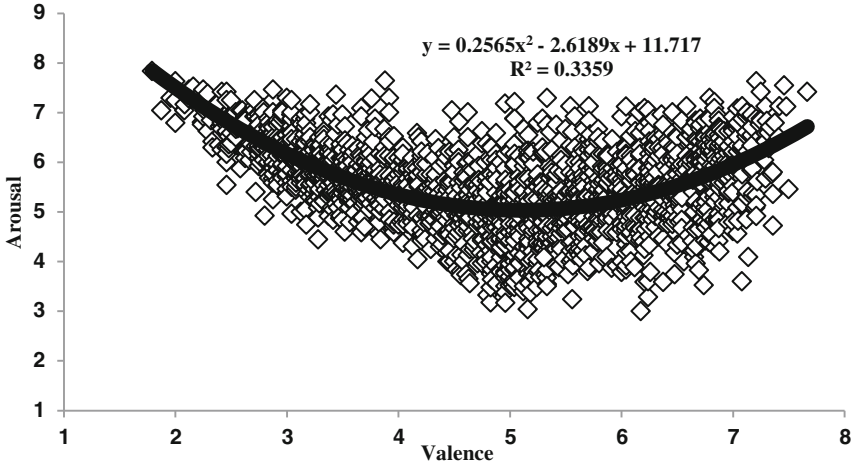


Fig. 1. Distribution of the mean ratings for VA pairs.

Relationship between valence and word frequency

Some previous studies have established that these two variables are slightly related [13]. In our dataset, their relations are plotted in Fig. 2 using a regression with the mean valence and its square as dependent variables and the mean word log frequency as an independent variable were conducted. Both the quadratic [$F(2, 2073) = 6.87, p = .001$] and the linear [$F(1, 2074) = 9.96, p = .002$] models are not reliable. Thus, we did not find any significant correlation between word frequency and valence.

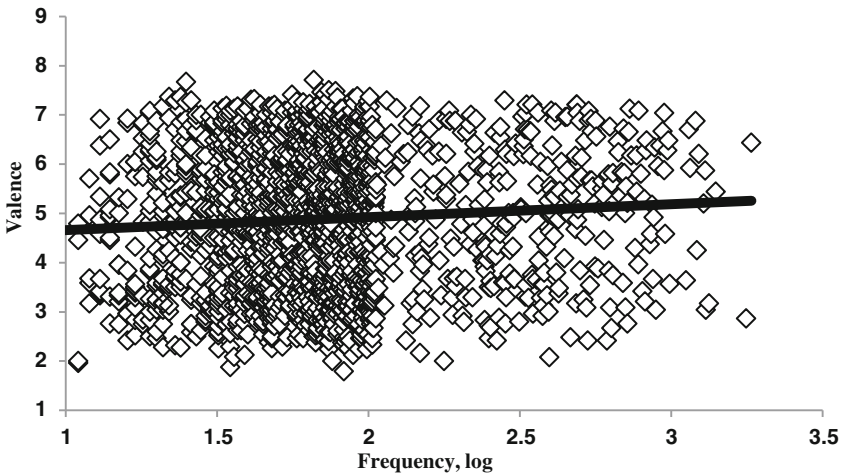


Fig. 2. Relationship between word frequency and valence.

4.5 2-Character Units in 4-Character Words

To further investigate the relationships among 4-character words, we examined the 2-character units from 434 pairs of 4-character words. These 2-character units are two consecutive characters in these 4-character words. As shown in Table 2, these 4-character words can be divided into 5 types according to the valence difference between the two 4-character words³. In the first type, the valence difference ranges from -3.99 to -0.92 with significant differences ($p < .001$). For example⁴, the word 滔天 (i.e., billowy, valence = 3.67) is both in the negative word 罪恶滔天 (i.e., monstrous crimes, valence = 1.74) and neutral word 波浪滔天 (i.e., a giant wave, valence = 5.08). In the second type, the valence difference ranges from $-.90$ to $-.21$ with significant differences ($p < .001$). For example, the word 攸关 (i.e., related, valence = 4.33) is both in the negative word 性命攸关 (i.e., life and death, valence = 3.83) and neutral word 利害攸关 (i.e., interests related, valence = 4.72). In the third type, the valence difference ranges from $-.20$ to 0.20 among without significant differences ($p = .88$). For example, the word 白费 (i.e., waste, valence = 2.33) is both in the negative word 白费心思 (i.e., to bother one's head for nothing, valence = 3.16) and negative word 白费口舌 (i.e., waste one's words, valence = 3.02). In the fourth type, the valence difference ranges from $.20$ to $.82$ with significant differences ($p < .001$). For example, the word 十年 (i.e., ten years, valence = 6.0) is both in the positive word 十年树木 (i.e., a wide-ranging project over many years, valence = 6.29) and the negative word 十年寒窗 (i.e., persevere ten years in one's studies in spite of hardships, valence = 5.48). In the fifth type, the valence difference ranges is from $.91$ to 4.22 with significant differences ($p < .001$). For example, the word 相待 (i.e., treat each other, valence = 5.67) is both in the positive word 坦诚相待 (i.e., be honest to each other, valence = 7.10) and negative word 冷眼相待 (i.e., sarcastic to each other valence = 2.96). These results indicate that the same 2-character units with different characters (i.e., context) may form various emotion words. In other words, these results indicate that the affective meaning of the 4-character words are worthy of separate study and annotation rather than taking them as compositions of their components.

³ Arousal differences are not reported in this section since the yare not significant.

⁴ The VA values of these 2-characters units for the examples in Table 2 were annotated by three other native Chinese speakers.

Table 2. Mean valence of 4-character words with the same 2-character units

Type	No.	Examples	Mean Valence	<i>t</i>	<i>p</i>
1	43	罪恶滔天	3.80	-14.4	<.001
	43	波浪滔天	5.79		
2	90	性命攸关	4.82	-3.1	<.001
	90	利害攸关	5.28		
3	159	白费心思	4.86	-.15	.88
	159	白费口舌	4.86		
4	92	十年树木	5.48	23.2	<.001
	92	十年寒窗	5.05		
5	50	坦诚相待	5.93	15.2	<.001
	50	冷眼相待	4.22		

In principle, a 4-character word is either compositional or non-compositional. If it is non-compositional, annotation is needed to supplement current 2-character affective lexicons. If it is compositional, the annotated 4-character lexicon can be used by a compositional model to infer the valence-arousal ratings of 4-character words from its component words. For example, the VA ratings of its component words 冷眼 and 相待 are (1.7, 5.0) and (5.7, 5.0) respectively based on three native Chinese speakers. The VA rating of the word 冷眼相待 is (2.8, 5.7), which is inferred according to certain compositional model.

5 Conclusions

The knowledge resource for Chinese only has 2-character words in the multi-dimensional affective space. This paper presents a large affective lexicon of 2,076 Chinese 4-character words obtained through psycho-linguistic annotations in controlled environment on valence and arousal. The data is cleaned to ensure high quality output. Analysis to the annotated data indicates that valence and arousal fit the classical U-shaped distribution. Most importantly, analysis to the data indicates that the same 2-character word that appears in different 4-character words can indeed show distinct affective meanings which implies that the affective meaning of 4-character words may not be computed by its component words. This dataset can be used for affective computing for text in the continuous valence-arousal space. It can also be used to study compositionality to infer valence-arousal of larger text units from that of the component words.

Acknowledgement. This project is supported partially by the CAS Key Laboratory of Mental Health (No. KLMH2014ZG14), the Hong Kong Scholars Program (No. XJ2015050), the National Natural Science Foundation of China (No. 31600887), RGC Funding (Pol- yU152006/16E), and HK Polytechnic University (PolyU RTVU and CERG PolyU 15211/14E).

References

1. Barrett, L.F., Mesquita, B., Ochsner, K.N., Gross, J.J.: The experience of emotion. *Ann. Rev. Psychol.* **58**, 373–403 (2007)
2. Schauenburg, G., Ambrasat, J., Schröder, T., vonScheve, C., Conrad, M.: Emotional connotations of words related to authority and community. *Behav. Res. Meth.* **47**(3), 720–735 (2015)
3. Russell, J.A.: Core affect and the psychological construction of emotion. *Psychol. Rev.* **110**(1), 145–172 (2003)
4. Ambrasat, J., von Scheve, C., Conrad, M., Schauenburg, G., Schröder, T.: Consensus and stratification in the affective meaning of human sociality. *Proc. Nat. Acad. Sci.* **111**(22), 8001–8006 (2014)
5. Scott, G.G., O'Donnell, P.J., Sereno, S.C.: Emotion words and categories: evidence from lexical decision. *Cognit. Process* **15**, 209–215 (2014)
6. Stone, P.J., Dunphy, D.C., Smith, M.S., Ogilvie, D.M.: *The General Inquirer: A Computer Approach to Content Analysis*. MIT Press, Cambridge, MA (1966)
7. Mehrabian, A.: Framework for a comprehensive description and measurement of emotional states. *Genet. Soc. Gen. Psychol. Monogr.* **121**, 339–361 (1995)
8. Ohana, B., and Tierney, B.: Sentiment classification of reviews using SentiWordNet. In: 9th IT Conference, Dublin Institute of Technology, Dublin, Ireland, pp. 1–9, 22–23 October 2009
9. Baroni, M., Zamparelli, R.: Nouns are vectors, adjectives are matrices: representing adjective-noun constructions in semantic space. In: *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pp. 1183–1193 (2010)
10. Mitchell, J., Lapata, M.: Composition in distributional models of semantics. *Cognit. Sci.* **34**(8), 1388–1429 (2010)
11. 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)*, pp. 1631–1642 (2013)
12. Bradley, M.M., Lang, P.J.: *Affective norms for English words (ANEW): Instruction manual and affective ratings*, Technical report C-1, the center for research in psychophysiology. University of Florida, Gainesville (1999)
13. Warriner, A.B., Kuperman, V., Brysbaert, M.: Norms of valence, arousal, and dominance for 13,915 English lemmas. *Behav. Res. Meth.* **45**(4), 1191–1207 (2013)
14. Schmidtke, D.S., Schröder, T., Jacobs, A.M., Conrad, M.: ANGST: Affective norms for German sentiment terms, derived from the affective norms for English words. *Behav. Res. Meth.* **46**(4), 1108–1118 (2014)
15. Monnier, C., Syssau, A.: Affective norms for 720 French words rated by children and adolescents (FANchild). *Behav. Res. Meth.* **49**, 1882–1893 (2017)
16. Stadthagen-Gonzalez, H., Imbault, C., Sánchez, M.A.P., Brysbaert, M.: Norms of valence and arousal for 14,031 Spanish words. *Behav. Res. Meth.* **49**, 111–123 (2017)
17. Yao, Z., Wu, J., Zhang, Y., Wang, Z.: Norms of valence, arousal, concreteness, familiarity, image ability, and context availability for 1,100 Chinese words. *Behav. Res. Meth.* **49**, 1374–1385 (2017)
18. Yu, L.-C., Lee, L.-H., Hao, S., Wang, J., He, Y., Hu, J., Lai, K.R., Zhang, X.: Building Chinese affective resources in valence-arousal dimensions. In: *Proceedings of NAACL-HLT*, pp. 540–545 (2016)
19. Wang, Y.N., Zhou, L.M., Luo, Y.J.: The pilot establishment and evaluation of Chinese affective word system. *Chin. Mental Health J.* **22**, 39–43 (2008)

20. Chinese Lexicon. Produced by State Key Laboratory of Intelligent Technology and Systems. Tsinghua University and Institute of Automation, Chinese Academy of Sciences. Retrieved from Chinese linguistic Data Consortium Beijing, China (2003)
21. Lexicon of common words in contemporary Chinese (protocol), Produced by Lexicon of common words in contemporary Chinese research team, The Commercial Press, Beijing, China (2008)
22. Mehrabian, A.: Pleasure-arousal-dominance: A general framework for describing and measuring individual differences in temperament. *Curr. Psychol.* **14**(4), 261–292 (1996)
23. Heise, D.R.: Semantic differential profiles for 1,000 most frequent English words. *Psychol. Monogr. Gen. Appl.* **79**(8), 1–31 (1965)
24. Hutto, C.J., Gilbert, E.: Vader: A parsimonious rule-based model for sentiment analysis of social media text. In: Eighth International AAAI Conference on Weblogs and Social Media, pp. 216–225 (2014)
25. Heise, D.R.: Affect control theory: concepts and model. *J. Math. Sociol.* **13**(1–2), 1–33 (1987)
26. Li, M., Lu, Q., Long, Y.: Representation learning of multiword expressions with compositionality constraint. In: Li, G., Ge, Y., Zhang, Z., Jin, Z., Blumenstein, M. (eds.) KSEM 2017. LNCS (LNAI), vol. 10412, pp. 507–519. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63558-3_43
27. Frege, G.: *The Foundations of Arithmetic: A Logico-Mathematical Enquiry into the Concept of Number*. Northwestern University Press, Evanston (1980)
28. Zhao, Y., Liu, Z., Sun, M.: Phrase type sensitive tensor indexing model for semantic composition. In: AAAI, pp. 2195–2202 (2015)
29. Calvo, R.A., Mac Kim, S.: Emotions in text: dimensional and categorical models. *Comput. Intell.* **29**(3), 527–543 (2013)
30. Moors, A., De Houwer, J., Hermans, D., Wanmaker, S., Van Schie, K., Van Harmelen, A.-L., De Schryver, M., De Winne, J., Brysbaert, M.: Norms of valence, arousal, dominance, and age of acquisition for 4,300 Dutchwords. *Behav. Res. Meth.* **45**(1), 169–177 (2013)

Statistical Machine Translation



Integrating Specialized Bilingual Lexicons of Multiword Expressions for Domain Adaptation in Statistical Machine Translation

Nasredine Semmar^(✉) and Meriama Laib

CEA, LIST, Vision and Content Engineering Laboratory, 91191 Gif-sur-Yvette, France
{nasredine.semmar,meriama.laib}@cea.fr

Abstract. Domain adaptation consists in adapting Machine Translation (MT) systems designed for one domain to work in another. Multiword expressions generally characterize specific-domains vocabularies. Translating multiword expressions is a challenge for current Statistical Machine Translation (SMT) systems because corpus-based approaches are effective only when large amounts of parallel corpora are available. However, parallel corpora are only available for a limited number of language pairs and domains, and the process of building corpora for several language pairs and domains is time consuming and expensive. This paper describes an experimental evaluation of the impact of using a specialized bilingual lexicon of multiword expressions in order to obtain better domain adaptation for the state of the art statistical machine translation system Moses. Our study concerns the English-French language pair and two kinds of texts: in-domain texts from Europarl (European Parliament Proceedings) and out-of-domain texts from Emea (European Medicines Agency Documents). We introduce three methods to integrate extracted bilingual multiword expressions in Moses. We experimentally show that integrating specialized bilingual lexicons of multiword expressions improve translation quality of Moses for both in-domain and out-of-domain texts.

Keywords: Statistical machine translation · Domain adaptation
Bilingual lexicon · Multiword expression

1 Introduction

A MultiWord Expression (MWE) is a combination of words for which syntactic or semantic properties of the whole expression cannot be obtained from its parts [1]. They constitute an important part of the lexicon of any natural language. MWEs play a vital role in several natural language processing applications such as Machine Translation (MT) and Cross-Language Information Retrieval (CLIR) because they often characterize specific-domains vocabularies. Word alignment approaches are used generally for building bilingual lexicons or translation models from parallel corpora. The performance of corpus-based machine translation approaches depends on the quality, quantity and proper domain matching of the training data. Translation models trained on a general domain would not work well in a technical domain. In several domains, available corpora

are not sufficient to make Statistical Machine Translation (SMT) approaches operational. In this paper, we discuss the application of domain adaptation to the state of the art statistical machine translation system Moses¹. In particular, our investigation focuses on the impact of using a domain-specific bilingual lexicon of MWEs on the performance of this system. Two kinds of texts corpora are used in our investigation: in-domain texts from Europarl (European Parliament Proceedings) and out-of-domain texts from Emea (European Medicines Agency Documents).

The remainder of the paper is organized as follows. We first survey previous work on domain adaptation for SMT in Sect. 2, then, we define in Sect. 3 the notion of Multi-Word Expression and describe different types of MWEs with examples. In Sect. 4, we describe a hybrid approach to build bilingual lexicons of multiword expressions from parallel corpora. Section 5 shortly presents the factored translation model of the open source SMT system Moses and the word alignment tool Giza++ which we consider as a baseline in this study. In Sect. 6, the experimental results in terms of BLEU scores are reported and discussed. Finally, the conclusion and future work are presented in Sect. 7.

2 Related Work

Domain adaptation consists in adapting MT systems designed for one domain to work in another. In recent years, several works have investigated how to adapt MT systems to a specific domain or topic [2–4]. Langlais [5] integrated domain-specific lexicons in the translation model of a SMT engine which yields a significant reduction in word error rate. Lewis et al. [6] developed domain specific SMT by pooling all training data into one large data pool, including as much in-domain parallel data as possible. They trained highly specific language models on in-domain monolingual data in order to reduce the dampening effect of heterogeneous data on quality within the domain. Hildebrand et al. [7] used an approach which consisted essentially in performing test-set relativization (choosing training samples that look most like the test data) to improve the translation quality when changing the domain. Civera and Juan [8] and Bertoldi and Federico [9] used monolingual corpora to adapt MT systems designed for Parliament domain to work in News domain. The obtained results showed significant gains in performance. Banerjee et al. [10] combined two separate domain models. Each model is trained from small amounts of domain-specific data. This data is gathered from a single corporate website. The authors used document filtering and classification techniques to realize the automatic domain detection. Daumé and Jagarlamudi [11] used dictionary-mining techniques to find translations for unseen words from comparable corpora and they integrated these translations into a statistical phrase-based translation system. They reported improvements in translation quality (between 0.5 and 1.5 BLEU points) on four domains and two language pairs. Pecina et al. [12] exploited domain-specific data acquired by domain-focused web crawling to adapt general-domain SMT systems to new domains. They observed that even small amounts of in-domain parallel data are more important for translation quality than large amounts of in-domain monolingual data. Wang et al. [13] used a single translation model and generalized a single-domain decoder to deal

¹ <http://www.statmt.org/moses/>.

with different domains. They used this method to adapt large-scale generic SMT systems for 20 language pairs in order to translate patents. The authors reported a gain of 0.35 BLEU points for patent translation and a loss of only 0.18 BLEU points for generic translation. Hasler et al. [14] studied empirically the effect of combining domain adaptation and topic adaptation within the same translation system. They reported that the best combined model yields BLEU improvements of up to 1.67 over an unadapted baseline system and 0.82 over a domain-adapted system.

Most of the approaches previously described for domain adaptation in statistical machine translation use Giza++² to build the translation model of the SMT system to be adapted. Giza++ produces translation tables, which contain both single words and multiword expressions. This alignment tool does not have a specific treatment for multiword expressions.

The approach used for domain adaptation in this study is close in spirit to the work of [5, 12]. It consists in adding a domain-specific bilingual lexicon of multiword expressions (extracted from a parallel domain-specific corpus) to the general-purpose training data of the SMT system.

3 Multiword Expressions

In Natural Language Processing (NLP), a multiword expression refers to a non-compositional sequence of words whose exact and unambiguous meaning, connotation and syntactic properties cannot be derived from the meaning or connotation of its components [1]. MWEs are frequently used in written texts and constitute a significant part of the language lexicon. Sag et al. [1] classify multiword expressions into two main categories: lexicalized phrases and institutionalized phrases (Fig. 1). Lexicalized phrases

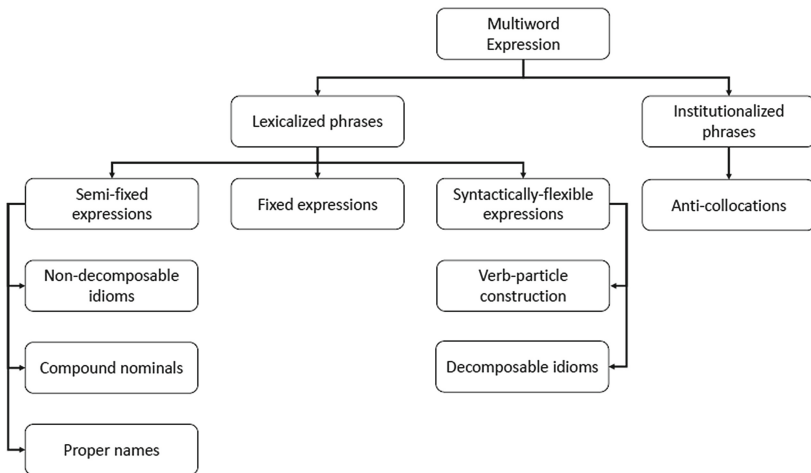


Fig. 1. Typology of multiword expressions.

² <http://www.statmt.org/moses/giza/GIZA++.html>.

“have at least partially idiosyncratic syntax or semantics, or contain “words” which do not occur in isolation”. Institutionalized phrases are “semantically and syntactically compositional, but statistically idiosyncratic”.

3.1 Lexicalized Phrases

In a decreasing order of lexical rigidity, lexicalized phrases are broken down into three classes: fixed expressions, semi-fixed expressions and syntactically-flexible expressions.

Fixed expressions are non-compositional sequences of words. They are syntactically and morphologically rigid and undergo neither internal modification nor morphological and syntactical variations (e.g. “nest of vipers” in English or “pomme de terre” in French). To determine whether or not a sequence of words is a fixed expression, we can use linguistic criteria such as using synonyms or adding words between its components (e.g. “nest of many black vipers” in English or “pomme de jolie terre lointaine” in French). Fixed expressions can be considered as single entries in the dictionary.

A semi-fixed expression is a non-compositional sequence of words whose components do not contribute to its figurative meaning. Semi-fixed expressions should respect a strict word order and some of them undergo limited lexical and morphological variability such as inflection and some variation in the reflexive form. According to their characteristics, they can be broken down into three basic categories: non-decomposable idioms, proper names and some compound nominals [1].

Non-Decomposable Idioms do not undergo syntax variability but their components accept lexical changes such as pronominal reflexivity form (e.g. “wet himself”, “wet themselves”), verbal inflection (“kick the bucket”, “kicked the bucket”) or passivization (e.g. “briser le silence” or “le silence est brisé” in French). Proper Names “are syntactically highly idiosyncratic” [1]. They can be complex with two or three proper names as components, including person, place and organization names. Compound Nominals are syntactically unalterable and undergo number inflection (e.g. “car park(s)” in English or “pomme(s) de terre” in French).

Unlike semi-fixed expressions, syntactically-flexible expressions undergo a wide degree of syntactic variation such as passivization (e.g. “The cat was let out of the bag”) and allow external elements to intervene between their components (e.g. “slow the car down”). This type of expressions includes verb-particle constructions, decomposable idioms. Particle verbs constructions are made up of a verb whose meaning is modified by one or more particles. They can be either semantically idiosyncratic such as “brush up on” or compositional such as “take after”, “look out”, “go back” and “run over”. Decomposable idioms tend to be syntactically flexible to some degree that is unpredictable. Semantically, they behave as if their components were linked parts contributing independently to the figurative interpretation of the expression as a whole.

3.2 Institutionalized Phrases

Institutionalized phrases are semantically and syntactically fully compositional, but statistically idiosyncratic [1]. They occur in a high frequency and their idiosyncrasy is

statistical rather than linguistic. They generally allow one available meaning. Institutionalized phrases often refer to “collocations”, described as sequences of words that statistically have a high probability to appear together whether they are contiguous or not (e.g. “make a difference”).

4 Building Specialized Bilingual Lexicons of Multiword Expressions from Parallel Corpora

There are mainly two strategies to extract bilingual MWEs from parallel corpora. The first strategy consists to acquire translations of phrases from parallel corpora in one step. Phrases are not necessary MWEs, they could be contiguous sequences of a few words that encapsulate enough context to be translatable [15]. The second strategy firstly, identifies monolingual MWEs candidates and then applies alignment approaches to find bilingual correspondences [16–18]. In the second strategy, MWEs extraction can be processed by using symbolic methods based on morpho-syntactic patterns, or, through statistical approaches, which use automatic measures to rank MWEs candidates. Finally, MWEs extraction can be done by using hybrid approaches, which combine the two first strategies.

Our approach for MWEs alignment is hybrid and consists first in identifying the relevant word groups through the use of n-gram statistics in both the source and target languages. Then for each source extracted MWE, it compiles a list of candidate translations through the use of two distance metrics. The list of candidates is then pruned using heuristics and morpho-syntactic patterns [19, 23].

4.1 Monolingual Extraction of MWEs

The role of this step consists to identify all the n-grams (up to 6-g) that may represent a MWE. This is done through frequency analysis and heuristic scoring. This step outputs two lists of MWEs, which we will refer to as SC (MWE in the Source language) and TC (MWE in the Target Language).

4.2 Frequency Distance Calculation Between MWEs

This step calculates for all source MWEs in SC the distance to each of the target MWEs in TC. The main idea of this metric is that if two MWEs are translations of each other then they must appear together in the corpus segments, and only together. Their frequency distance is then calculated as follows:

$$FD(s, t) = \frac{|f(s) - f(t)|}{\max(f(s), f(t))}$$

Where s corresponds to a SL (Source Language) MWE, t corresponds to a TL (Target Language) MWE, $f(s)$ is the frequency of the source MWE, and $f(t)$ is frequency of the target MWE.

As we can see, if t is the translation of s , $f(s) = f(t)$ and we have 0 distance. Also, if two MWEs always occur together but one is much more frequent than the other, the distance reaches 1 and they are not considered translations of each other. We have chosen to apply a threshold of 0.25 as the maximum allowable distance. This threshold can be tuned to achieve better precision. It is calculated empirically after analyzing a small aligned corpus.

4.3 Co-occurrence Distance Calculation Between MWEs

The previous step only considers frequencies so it may be possible for two completely unrelated MWEs to achieve a low distance score. However we also check for a co-occurrence score which allows the rejection of the MWEs that fortuitously have similar frequency. Since they would not appear in the same segments, the value of $X_i - Y_i$ would increase. The candidate list can be ordered through CD .

$$CD(X_i, Y_i) = \frac{\sqrt{\sum (X_i - Y_i)^2}}{N}$$

Where X_i is the number of occurrences of the MWE s in the i^{th} segment of the SL, Y_i is the number of occurrences of the MWE t in the i^{th} segment of the TL, and N is the number of segments.

4.4 Pruning MWEs Candidates

After obtaining an ordered list of target MWEs candidates, we filter the results by removing:

- The candidates whose length is too different from the source MWE.
- The candidates who have been previously aligned with another source MWE and where the co-occurrence score was better.

5 Baseline MWEs Alignment Tool

In order to study the impact of MWEs on the quality of translation, we used the factored translation model³ of the SMT system Moses as our baseline. It is an extension of the phrase-based model which is limited to the mapping of phrases without any explicit use of linguistic information. The factored model enables the use of additional markup at the word level (Fig. 2).

³ <http://www.statmt.org/moses/?n=Moses.FactoredModels>.

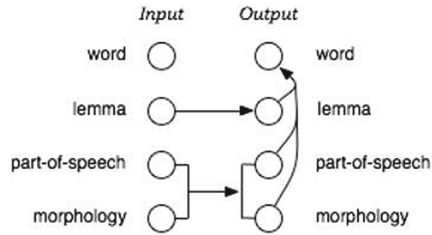


Fig. 2. Factored model used in the SMT baseline system Moses.

Our model operates on lemmas instead of surface forms, in which the translation process is broken up into a sequence of mapping steps:

1. Translate the lemmas of the source language (Input) into lemmas in the target language (Output).
2. Generate surface forms given the lemma and linguistic features (morphology and part-of-speech).

In Moses, translation models are produced by the word alignment tool Giza++ which we consider in this study as our baseline.

6 Experimental Results

The quality of alignment of MWEs and the impact of using MWEs in machine translation have been evaluated, firstly, manually, by comparing the results of our approach and the baseline (Giza++) with a reference alignment; and secondly, automatically, by comparing the performance of the SMT system Moses after integrating the specialized bilingual lexicon produced by our MWEs alignment approach into its translation model.

6.1 Manual Evaluation

Our approach for MWEs alignment and Giza++ have been evaluated using the metrics described in [20]. The corpus used to evaluate the performance of the English-French MWEs aligners is composed of a set of 1992 parallel sentences extracted from Europarl (European Parliament Proceedings) and lemmatized by the multilingual analyzer LIMA [21]. This parallel corpus is composed of 46265 English words and 49332 French words and has been used to build manually the reference alignment by the Yawat tool [22]. Alignment with Giza++ was achieved in source–target and target–source directions and the results were merged using the union heuristic.

Table 1 summarizes the results of our approach for English–French MWEs alignment and the baseline (Giza++) in terms of precision, recall and f-measure. These results show that our approach performs slightly better than Giza++ but the f-measure remains low compared to single-word alignment approaches [19].

Table 1. Performances of Giza++ and our approach for aligning MWEs.

MWEs aligner	Precision	Recall	F-measure
Baseline (Giza++)	0.83	0.37	0.51
Our approach	0.81	0.39	0.52
Our approach + filtering with morpho-syntactic patterns	0.89	0.38	0.53

Because of the statistical nature of our approach, it is therefore not easy to improve the “Recall”. However, we can improve the “Precision” by using morpho-syntactic patterns to keep only specific strings and filter out undesirable ones such as candidates composed mainly of stop words [23]. We built a dozen of morpho-syntactic patterns (Table 2) and we applied them to filter the MWEs produced in step 4.

Table 2. Some English and French filtering morpho-syntactic patterns.

Pattern	English MWE	French MWE
Adj-Noun	planery meeting	libre circulation
Noun-Prep-Noun	point of view	point de vue
Noun-Prep-Adj-Noun	court of first instance	court de première instance
Noun-Noun	member state	état membre

As we can see in Table 1, the filtering morpho-syntactic patterns have certainly improved the precision by 8% (from 0.81 to 0.89) but the recall has dropped by 1% (from 0.39 to 0.38). Table 3 illustrates some aligned MWEs.

Table 3. Some aligned MWEs.

English MWE	French MWE
express a opinion	exprimer un avis
order of business	ordre du jour
extension of the Monday sitting	prolongation de le séance de lundi
simply do not allow	ne permettre absolument pas

After analyzing some alignments of MWEs, we noticed that our approach augmented with morpho-syntactic patterns:

- allows the alignment of MWEs based on a verb “express a opinion/exprimer un avis”,
- captures the negative form in equivalent expressions such as “simply do not allow” and “ne permettre absolument pas”.

6.2 Automatic Evaluation Through a Translation Task

The unavailability of a reference alignment of a significant size for MWEs does not allow us to achieve a large evaluation and to compare our approach with the state-of-the-art work. That is why we decided to study the impact of MWEs on the quality of translation by integrating the results of our word alignment approach in the translation

model of Moses. The goal is to study in what respect MWEs are useful to improve the performance of this SMT system.

Data and Experimental Setup

In order to study the impact of using a domain-specific bilingual lexicon on the performance of Moses, we conducted our experiments on two English-French parallel corpora (Table 4): Europarl (European Parliament Proceedings) and Emea (European Medicines Agency Documents). Both corpora were extracted from the open parallel corpus OPUS [24]. Since we use the factored translation model of the open source statistical machine translation system Moses, we lemmatized the parallel corpus before applying our MWEs alignment approach using the multilingual analyzer LIMA [21].

Table 4. Corpora used to train Moses language and translation models (K refers to 1000).

Run no.	Training (# sentences)	Tuning (# sentences)
1	150K + 10K (Europarl + Emea)	2K + 0.5K (Europarl + Emea)
2	150K + 20K (Europarl + Emea)	2K + 0.5K (Europarl + Emea)
3	150K + 30K (Europarl + Emea)	2K + 0.5K (Europarl + Emea)

Evaluation consists in comparing translation results produced by Moses on in-domain and out-of-domain texts. The English-French training corpus is used to build Moses’s translation and language models. We conducted three runs and two test experiments for each run: In-Domain and Out-Of-Domain. For this, we randomly extracted 500 parallel sentences from Europarl as an In-Domain corpus and 500 pairs of sentences from Emea as an Out-Of-Domain corpus. These experiments are done to show the impact of the domain vocabulary on the translation results. The domain vocabulary is identified by a bilingual lexicon of multiword expressions which is extracted automatically from the specialized parallel corpus (Emea) using our MWEs alignment approach.

We used the following three methods to integrate into the translation model of Moses the specialized bilingual lexicon which is built automatically by our MWEs alignment approach [23, 26–28]:

- **Moses_{CORPUS}**: In this method, we add the extracted bilingual lexicon as a parallel corpus and retrain the translation model. By increasing the occurrences of the specialized words and their translations, we expect a modification of alignment and probability estimation.
- **Moses_{TABLE}**: This method consists in adding the extracted bilingual lexicon of MWEs into the phrase table. We use the similarity measure provided by our word alignment approach for each specialized word of the bilingual lexicon as a translation probability.
- **Moses_{FEATURE}**: In this method, we extend “Moses_{TABLE}” by adding a new feature indicating whether a word comes from the specialized bilingual lexicon or not (1 or 0 is introduced for each entry of the phrase table). The idea is to guide Moses to choose bilingual MWEs instead of its phrases.

6.2.1 Results and Discussion

The performance of Moses is evaluated using the BLEU score [25] on the two test sets for the three runs described in the previous section. Note that we consider only one reference per sentence. We report respectively in Tables 5 and 6 the performance of Moses.

Table 5. BLEU scores of Moses for In-Domain texts.

Run no.	Giza++	Moses _{CORPUS}	Moses _{TABLE}	Moses _{FEATURE}
1	32.62	32.41	32.36	32.55
2	33.81	33.76	33.71	33.79
3	34.25	34.23	34.21	34.24

Table 6. BLEU scores of Moses for Out-Of-Domain texts.

Run no.	Giza++	Moses _{CORPUS}	Moses _{TABLE}	Moses _{FEATURE}
1	22.96	22.82	22.75	22.91
2	23.30	23.09	23.04	23.27
3	24.55	24.49	24.45	24.52

As shown in Tables 5 and 6, Moses achieves a relatively high BLEU score for In-Domain texts and the scores of Moses when using the Moses_{FEATURE} method are better in all the runs for both In-Domain and Out-Of-Domain texts. The Moses_{CORPUS} method (when compared to the Moses_{FEATURE} method) comes next with a drop of only 0.01 points for In-Domain sentences and 0.03 points for the Out-Of-Domain texts. Similarly, as we can see, filtering patterns have improved BLEU scores of the three integration methods in all the runs for Out-Of-Domain texts and the Moses_{FEATURE} method achieves the best performances (Table 7). These results confirm that adding specialized bilingual MWEs to the translation model of Moses improves its performance [26–29].

Table 7. BLEU scores of Moses for Out-Of-Domain texts when using our alignment approach with morpho-syntactic patterns.

Run no.	Giza++	Moses _{CORPUS}	Moses _{TABLE}	Moses _{FEATURE}
1	22.96	23.41	23.06	23.62
2	23.30	23.98	23.75	24.15
3	24.55	25.39	24.91	25.43

In order to show the impact of the domain vocabulary (represented by the bilingual lexicon of MWEs extracted with our word alignment approach), on the translation results of Moses, we manually analyzed an example of translations drawn from the Out-Of-Domain test corpus (Table 8). The first observation is that, in some cases, it is just impossible to perform a word-to-word alignment between two MWEs that are translation of each other. For example, the Moses_{FEATURE} method proposes the compound word “glycémie à jeun” as a translation for the expression “fasting blood glucose” which is correct, but, Moses_{CORPUS} and Moses_{TABLE} methods propose respectively the translations

Table 8. Translations produced by Moses for a sentence from the Emea corpus.

Input sentence (Emea)	in the 12 week acute phase of three clinical trials of duloxetine in patients with diabetic neuropathic pain, small but statistically significant increases in fasting blood glucose were observed in duloxetine-treated patients
Translation reference	lors de la phase aiguë de 12 semaines de trois essais cliniques étudiant la duloxétine chez les patients souffrant de douleur neuropathique diabétique, des augmentations faibles, mais statistiquement significatives de la glycémie à jeun ont été observées chez les patients sous duloxétine
Moses translation with Moses _{CORPUS} method	dans le 12 semaines de la phase aiguë trois études cliniques de duloxetine chez les patients avec douleur neuropathique diabétique, petites mais statistiquement significatif augmentations de répréhensible glycémie artérielle a été observée chez les patients traités duloxetine
Moses translation with Moses _{TABLE} method	dans le 12 semaine de la phase aiguë de trois essais cliniques de duloxetine dans les patients avec douleur neuropathique diabétique, petites mais statistiquement augmentations considérables dans le sang répréhensible glucose ont été constatées dans les patients duloxetine traités
Moses translation with Moses _{FEATURE} method	dans le 12 semaines de la phase aiguë de trois essais cliniques chez les patients avec douleur neuropathique diabétique, petites mais des augmentations statistiquement significatives de la glycémie à jeun ont été observées chez les patients traités duloxétine

“répréhensible glycémie artérielle” and “sang répréhensible glucose” which are completely wrong. However, all the integration methods translate correctly the multiword expressions “diabetic neuropathic pain/douleur neuropathique diabétique”, “acute phase/phase aiguë” and “clinical trials/essais cliniques”. It seems that the probabilities of the alignments proposed by Giza++ for these multiword expressions were very high and helped Moses decoder to choose these alignments. On the other hand, as we can see, all the translations have many spelling and grammatical errors, and in particular, the translations of some multiword expressions (statistically significant increases/statistiquement significatif augmentations, statistically significant increases/statistiquement augmentations considérables) produced by Moses_{CORPUS} and Moses_{TABLE} methods are very approximate. This result can be explained by the fact that, on the one hand, statistical machine translation toolkits like Moses have not been designed with grammatical error correction in mind, and on the other hand, Giza++ could produce errors in particular when it aligns multiword expressions [30].

For the multiword expression “duloxetine-treated patients”, the methods $Moses_{FEATURE}$ and $Moses_{CORPUS}$ provide a same translation which is more or less correct (duloxetine-treated patients/patients traités duloxetine). However, the method $Moses_{TABLE}$ provides a translation in a poor grammar (duloxetine-treated patients/patients duloxetine traités).

Finally on this point, we can observe that the major issues of Moses concern errors produced by Giza++ when aligning multiword expressions (translation model), incorrect spelling and poor grammar generated by the decoder (language model). To handle the first issue, we proposed to take into account the specialized bilingual lexicon extracted with the MWEs aligner into Moses’s phrase table and we added a new feature indicating whether a word comes from this lexicon or not ($Moses_{FEATURE}$ integration method). However, for spelling and grammar errors, Moses has no specific treatment.

7 Conclusion and Future Work

In this paper, we have presented a hybrid approach to build automatically bilingual lexicons of multiword expressions. In particular, we have described an experimental evaluation of the impact of using a specialized bilingual lexicon of multiword expressions in domain adaptation. We have more specifically used three strategies in order to integrate into the translation model of Moses the specialized bilingual lexicon. The obtained results have showed that the strategy based on adding a new feature indicating whether a word comes from the specialized bilingual lexicon or not, improves translation quality of Out-Of-Domain texts without loss of translation quality when translating In-Domain texts.

This study leaves two important open issues, which certainly deserve further research. First, we should explore machine learning approaches to extend the morpho-syntactic patterns to handle with other forms of MWEs. The second perspective is to explore the integration of bilingual MWEs into other machine translation models such as rule-based translation ones. We also expect to explore the use of LSTM (Long Short-Term Memory) neural network language models for rescoring the n-best translations produced by Moses.

Acknowledgments. This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 700381.

References

1. Sag, Ivan A., Baldwin, T., Bond, F., Copestake, A., Flickinger, D.: Multiword expressions: a pain in the neck for NLP. In: Gelbukh, A. (ed.) CICALing 2002. LNCS, vol. 2276, pp. 1–15. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45715-1_1
2. Bungum, L., Gambäck, B.: A survey of domain adaptation in machine translation towards a refinement of domain space. In: Proceedings of the India-Norway Workshop on Web Concepts and Technologies (2011)

3. Ceaşu, A., Tinsley, J., Zhang, J., Way, A.: Experiments on domain adaptation for patent machine translation in the PLuTO project. In: Proceedings of EAMT (2011)
4. Mathur, P., Federico, M., Köprü, S., Khadivi, S., Sawaf, H.: Topic adaptation for machine translation of E-commerce content. In: Proceedings of MT Summit XV (2015)
5. Langlais, P.: Improving a general-purpose statistical translation engine by terminological lexicons. In: Proceedings of COLING: Second International Workshop on Computational Terminology (2002)
6. Lewis, W.D., Wendt, C., Bullock, D.: Achieving domain specificity in SMT without overt siloing. In: Proceedings of LREC (2010)
7. Hildebrand, A.S., Eck, M., Vogel, S., Alex, W.: Adaptation of the translation model for statistical machine translation based on information retrieval. In: Proceedings of the EAMT (2005)
8. Civera, J., Juan, A.: Domain adaptation in statistical machine translation with mixture modelling. In: Proceedings of the Second Workshop on Statistical Machine Translation (2007)
9. Bertoldi, N., Federico, M.: Domain adaptation for statistical machine translation with monolingual resources. In: Proceedings of the 4th Workshop on Statistical Machine Translation (2009)
10. Banerjee, P., Du, J., Li, B., Naskar, S.K., Way, A., van Genabith, J.: Combining multi-domain statistical machine translation models using automatic classifiers. In: Proceedings of AMTA (2010)
11. Daumé III, H., Jagarlamudi, J.: Domain adaptation for machine translation by mining unseen words. In: Proceedings of ACL (2011)
12. Pecina, P., Toral, A., Way, A., Papa-vassiliou, V., Prokopidis, P., Giagkou, M.: Towards using web-crawled data for domain adaptation in statistical machine translation. In: Proceedings of EAMT (2011)
13. Wang, W., Macherey, K., Macherey, W., Och, F., Xu, P.: Improved domain adaptation for statistical machine translation. In: Proceedings of AMTA (2012)
14. Hasler, E., Haddow, B., Koehn, P.: Combining domain and topic adaptation for SMT. In: Proceedings of AMTA (2014)
15. DeNero, J., Klein, D.: The complexity of phrase alignment problems. In: Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies (2008)
16. Daille, B., Gaussier, E., Langé, J.M.: Towards automatic extraction of monolingual and bilingual terminology. In: Proceedings of the 15th Conference on Computational Linguistics ACL (1994)
17. Blank, I.: Terminology extraction from parallel technical texts. In: Véronis, J. (ed.) *Parallel Text Processing*, vol. 13. Springer, Dordrecht (2000). https://doi.org/10.1007/978-94-017-2535-4_12
18. Barbu, A.M.: Simple linguistic methods for improving a word alignment algorithm. In: Proceedings of the 7th International Conference on the Statistical Analysis of Textual Data (2004)
19. Semmar, N., Servan, C., De Chalendar, G., Le Ny, B., Bouzaglou, J.J.: A hybrid word alignment approach to improve translation lexicons with compound, words and idiomatic expressions. In: Proceedings of the 32nd Translating and the Computer Conference, ASLIB (2010)
20. Mihalcea, R., Pedersen, T.: An evaluation exercise for word alignment. In: Proceedings of the HLT-NAACL 2003 Workshop on Building and Using Parallel Texts: Data Driven Machine Translation and Beyond (2003)

21. Besançon, R., De Chalendar, G., Ferret, O., Gara, F., Laib, M., Mesnard, O., Semmar, N.: LIMA: a multilingual framework for linguistic analysis and linguistic resources development and evaluation. In: Proceedings of LREC (2010)
22. Germann, U.: Yawat: yet another word alignment tool. In: Proceedings of ACL 2008
23. Bouamor, D., Semmar, N., Zweigenbaum, P.: Identifying bilingual Multiword expressions for statistical machine translation. In: Proceedings of LREC (2012)
24. Tiedemann, J.: Parallel data, tools and interfaces in OPUS. In: Proceedings of LREC (2012)
25. Papineni, K., Roukos, S., Ward, T., Zhu, W.: BLEU: a method for automatic evaluation of machine translation. In: Proceedings of ACL (2002)
26. Semmar, N., Zennaki, O., Laib, M.: Improving the performance of an example-based machine translation system using a domain-specific bilingual lexicon. In: Proceedings of the 29th Pacific Asia Conference on Language, Information and Computation, PACLIC (2015)
27. Semmar, N., Zennaki, O., Laib, M.: Evaluating the impact of using a domain-specific bilingual lexicon on the performance of a hybrid machine translation approach. In: Proceedings of Recent Advances in Natural Language Processing International Conference, RANLP (2015)
28. Bouamor, D., Semmar, N., Zweigenbaum, P.: Automatic construction of a multiword expressions bilingual lexicon: a statistical machine translation evaluation perspective. In: Proceedings of the 3rd Workshop on Cognitive Aspects of the Lexicon, COLING (2012)
29. Ren, Z., Lu, Y., Cao, J., Liu, Q., Huang, Y.: Improving statistical machine translation using domain bilingual multiword expressions. In: Proceedings of the Workshop on Multiword Expressions, ACL-IJCNLP (2009)
30. Fraser, A., Marcu, D.: Measuring word alignment quality for statistical machine translation. *Assoc. Comput. Linguist.* **33**(3), 293–303 (2007)



Logical Parsing from Natural Language Based on a Neural Translation Model

Liang Li¹, Yifan Liu¹, Zengchang Qin¹(✉), Pengyu Li², and Tao Wan³

¹ Intelligent Computing and Machine Learning Lab School of ASEE,
Beihang University, Beijing 100191, China
zcqin@buaa.edu.cn

² School of Mechanical Engineering and Automation,
Beihang University, Beijing 100191, China

³ School of Biological Science and Medical Engineering,
Beihang University, Beijing 100191, China
taowan@buaa.edu.cn

Abstract. Semantic parsing has emerged as a powerful paradigm for natural language interface and question answering systems. Traditional methods of building a semantic parser rely on high-quality lexicons, hand-crafted grammars and linguistic features which are limited by applied domain or representation. In this paper, we propose an approach to learn from denotations based on the Seq2Seq model augmented with attention mechanism. We encode input sequence into vectors and use dynamic programming to infer candidate logical forms. We utilize the fact that similar utterances should have similar logical forms to help reduce the searching space. Through learning mechanism of the Seq2Seq model, we can learn mappings gradually with noises. Curriculum learning is adopted to make the learning smoother. We test our model on a small arithmetic domain which shows our model can successfully infer the correct logical forms and learn a meaningful semantic parser.

Keywords: Logical parsing · Neural language understanding
Seq2Seq · Attention model

1 Introduction

The problem of learning a semantic parser has been receiving significant attention. Semantic parsers map natural language into a logical form that can be executed on a knowledge base and return an answer (denotation). The early works use logical forms as supervision [7, 17, 18, 27], given a set of input sentences and their corresponding logical forms, learning a statistical semantic parser by weighting a set of rules mapping lexical items and syntactic patterns to their logical forms. Given an input, these rules can be applied recursively to derive

L. Li, Y. Liu—These authors contributed equally to this work and should be considered co-first authors.

the most probable logical form. However, the tremendous computation power needed for annotating logical forms has turned the trend to weak supervision – using denotation of logical forms as the training target. It has been successfully applied in different fields including question-answering [5, 11, 15, 23] and robot navigation [1]. All these works need hand-crafted grammars which are crucial in semantic parsing, but it is obviously an obstacle for generalization. Wang *et al.* [26] build semantic parsers in 7 different domains and hand-engineer a separate grammar for each domain.

The invention of the sequence-to-sequence (Seq2Seq) model [24] provides an alternative method to tackle the mapping problem and no more manual grammars are needed. The ability to deal with sequences with changeable length as input and/or output has translated this model into applications including machine translation [24], syntactic parsing [25], and question answering [9]. All of these works do not need hand-crafted grammars and are so-called end-to-end learning. But they do not resolve the problem of supervision by denotation, which makes one step further and needs logic reasoning and operation. Our model adopts the encoder-decoder framework and tries to use denotation as the target of supervised learning. We take the advantage of the Seq2Seq model’s ability of tackling input/output with different length and grammar-free form to learn the mappings from natural language to logical forms. Our main focus is to infer logical forms from denotations in a generalized way. We wish to add minimal extra constraints or manual features, so that it can be applied to other domains. For now, we can not infer correct logical forms all the time but we prove that the Seq2Seq model is capable of learning with noises in training data and the curriculum learning form mitigates this effect.

A problem of weak supervision is the search of the consistent logical forms when only denotation is available. The number of logical forms grows exponentially as their size increases, we may also obtain a large set of wrong logical forms when inferring from denotations [22]. To control the searching space, previous works relied on restricted sets of rules which limits expressivity and are possible to rule out the correct logical form. Instead, we utilize dynamic programming to construct the candidate logical form set with an extra base case set to filter all the candidates, and use the ongoing training Seq2Seq model to determine the best one. In this way, we can maintain the expressivity and train our model iteratively by feeding the suggested best logical form back to our model. Consequently, a right pick result in positive learning, which allows our model to put a higher probability on the correct logical form over others and leads to the desired mapping over training.

We evaluate our model on arithmetic domain through a toy example. And the model is capable of translating row sentences into mathematical equations in structured tree form and returning the answer directly. We provide a base case set which serves as a pivot for the model to learn. Our model learns the arithmetic calculation in a curriculum way, where simpler sentences with fewer words are inputted at initial state. Our model assumes no prior linguistic knowledge and learns the meaning of all the words, compositionality, and order of operations from just the natural language – denotation pairs.

1.1 Related Work

We adopt the general encoder-decoder framework based on neural networks augmented with attention mechanism [2], which allows the model to learn soft alignment between utterances and logical forms. Our work is related to [6] and [25], both of which use the Seq2Seq model to map natural language to logical forms in tree structure without hand-engineered features. But our work goes one step further by using denotation of logical form as the learning target and regard logical forms as latent variables.

How to reduce the searching space is a chief challenge in weak supervision. A common approach is to constrain the set of possible logical form compositions, which can significantly reduce the searching space but also constrain the expressivity [21]. Lao *et al.* [12] use random walks to generate logical forms and use denotation to cut down the searching space during learning. Liang *et al.* [14] adopt a similar method to narrow down the options and allow more complex semantics to be composed. Different from generating logical forms forwardly as mentioned above, an alternative method is to use denotation to infer logical forms using dynamic programming [13,22]. In this way, it is more likely to recover the full set and find the desired one. Inspired by their work, we also employ this method and store the denotation – logical form pairs in advance to accelerate the lookup efficiency.

Our work is similar to [16] in the sense that we both focus on the arithmetic domain and learn from denotations directly. But their work relies on hand-crafted grammars to construct logical forms and hand engineered features to filter out incorrect logical forms. Instead, the Seq2Seq model we use is grammar-free and the features we select to screen logical forms is more general. Our main idea is that similar utterance should have similar logical forms, so that we can filter out incorrect logical forms by using similarity measurement. We build up a small base case set to assist this idea and the similarity function we adopt is simply bag-of-words, which is replaceable when extending to other fields. Furthermore, to sort the candidates, we do not have a particular scoring function to weight extracted features, instead, we use the ongoing training Seq2Seq model to evaluate their loss.

There are some other related work of semantic parsing by using recurrent neural networks with other attention mechanism. For example, Neural Programmer [20] augmented with a small set of arithmetic and logic operations is able to perform complex reasoning in question answering [19]. Neural Turing Machines [8] can infer simple algorithms such as copying and sorting with external memory.

2 Sequence-to-Sequence Model

Before introducing the main idea, let us briefly describe the background of the Seq2Seq model [24] with attention mechanism.

2.1 Background

The Seq2Seq model takes a source sequence $X = (x_1, x_2, \dots, x_T)$ as input and outputs a translated sequence $Y = (y_1, y_2, \dots, y_{T'})$. The model maximizes the generation probability of Y conditioned on X : $p(y_1, \dots, y_{T'} | x_1, x_2, \dots, x_T)$. Specifically, the Seq2Seq is in an encoder-decoder structure. In this framework, an encoder reads the input sequence word by word into a vector c through recurrent neural network (RNN).

$$h_t = f(x_t, h_{t-1}) \quad (1)$$

and

$$c = q(h_1, \dots, h_T),$$

where h_t is the hidden state at time t , c is commonly taken directly from the last hidden state of encoder $q(h_1, \dots, h_T) = h_T$, and f is a non-linear transformation which can be either a long-short term memory unit (LSTM) [10] or a gated recurrent unit (GRU) [3]. In this paper, LSTM is adopted and is parameterized as

$$\begin{pmatrix} i_t \\ f_t \\ o_t \\ \tilde{c}_t \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} T \begin{pmatrix} x_t \\ h_{t-1} \end{pmatrix} \quad (2a)$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \tilde{c}_t \quad (2b)$$

$$h_t = o_t \circ \tanh(c_t) \quad (2c)$$

where \circ is an element-wise multiplication, T is an affine transformation, σ is the logistic sigmoid that restricts its input to $[0,1]$, i_t , f_t and o_t are the input, forget, and output gates of the LSTM, and c_t is the memory cell activation vector. The forget and input gates enable the LSTM to regulate the extent to which it forgets its previous memory and the input, while the output gate controls the degree to which the memory affects the hidden state. The encoder employs bidirectionality, encoding the sentences in both the forward and backward directions, an approach adopted in machine translation [2,4]. In this way, the hidden annotations $h_t = (\overrightarrow{h}_t^T; \overleftarrow{h}_t^T)$ concatenate forward \overrightarrow{h}_t^T and backward annotations \overleftarrow{h}_t^T together, each determined using Eq. 2c.

The decoder is trained to estimate generation probability of next word y_t given all the previous predicted words and the context vector c . The objective function of Seq2Seq can be written as

$$p(y_1, \dots, y_{T'} | x_1, \dots, x_T) = \prod_{t=1}^{T'} p(y_t | c, y_1, \dots, y_{t-1}). \quad (3)$$

Given a RNN, each conditional probability is modeled as

$$p(y_t | y_1, \dots, y_{t-1}, c) = g(y_{t-1}, s_t, c), \quad (4)$$

where g is a non-linear function that outputs the probability of y_t , y_{t-1} is the predicted word at time $t-1$ in the response sequence, and s_t is the hidden state of the decoder RNN at time t , which can be computed as

$$s_t = f(y_{t-1}, s_{t-1}, c). \quad (5)$$

2.2 Attention Mechanism

The traditional Seq2Seq model predicts each word from the same context vector c , which deprives the source sequence information and makes the alignment imprecisely. To address this problem, attention mechanism is introduced to allow decoder focusing on the source sequence instead of a compressed vector upon prediction [2]. In Seq2Seq with attention mechanism, each y_i in Y corresponds to a context vector c_i instead of c . The conditional probability in Eq. 4 becomes

$$p(y_i | y_1, \dots, y_{i-1}, x) = g(y_{i-1}, s_i, c_i), \quad (6)$$

where the hidden state s_i is computed by

$$s_i = f(y_{i-1}, s_{i-1}, c_i). \quad (7)$$

The context vector c_i is a weighted average of all hidden states $\{h_t\}_{t=1}^T$ of the encoder, defined as

$$c_i = \sum_{j=1}^T \alpha_{ij} h_j, \quad (8)$$

where the weight α_{ij} is given by

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^T \exp(e_{ik})}, \quad (9)$$

where e_{ij} is an alignment model which scores how well the inputs around position j and the output at position i match,

$$e_{ij} = a(s_{i-1}, h_j), \quad (10)$$

where a is a feed forward neural network, trained jointly with other components of the system.

3 Learning from Denotations

We use the triple $\langle u, s, d \rangle$ to denote the linguistic objects, where u is an utterance, s is a logical form and d is the denotation of s . We use $[u]$ to represent the translation of utterance into its logical form, and we use $\llbracket s \rrbracket$ for the denotation of logical form s . Each training data is composed by the pair $\langle u, d \rangle$ without explicitly telling its correct logical form s .

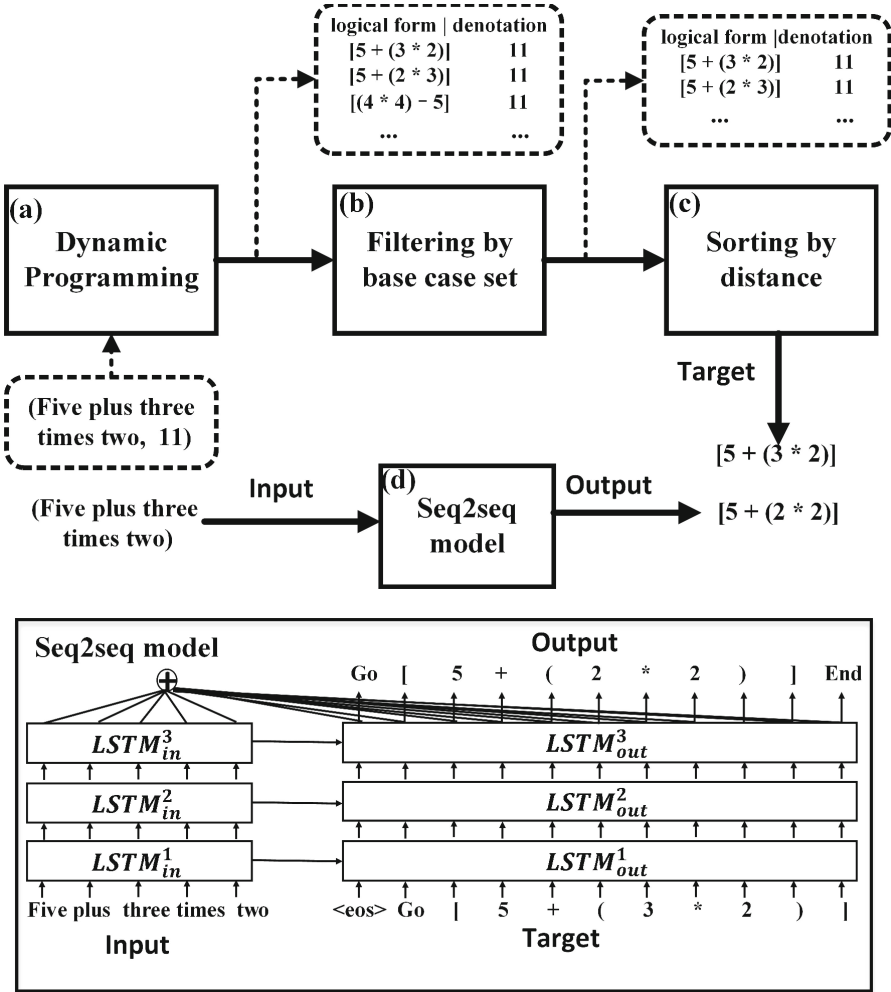


Fig. 1. Parsing the utterance “five plus three times two” with denotation 11. (a) Inferred logical forms from the denotation using dynamic programming. (b) Candidate logical forms are filtered by base case set. (c) Using the Seq2Seq model to rank the loss of each candidate logical form and return the least one for training. (d) The Seq2Seq model with attention mechanism with 3-layer LSTM on both encoder and decoder side.

With denotation as target label of learning, the Seq2Seq model is trained to put a high probability on $|u|$'s that are consistent-logical forms that execute to the correct denotation d . When the space of logical forms is large, searching for the correct logical form could be cumbersome. Additionally, different from the previous study which incorporates prior knowledge such as word embedding [14], object categories [1], our model in this mathematical expression learning

example has no such knowledge and has to learn the meanings of input utterance. Here we formally describe the methods to reduce the searching space and how to infer the correct logical forms from denotations.

3.1 Dynamic Programming on Denotations

Our first step is to generate all logical forms that have the correct denotations. Formally, given a denotation d , we wish to generate a candidate logical form set that satisfy the denotation demand $\Omega = \{s \llbracket s \rrbracket = d\}$. Previous work use beam search to generate candidates but it is hard to recover the full set Ω due to pruning. Noticing that one denotation may correspond to multiple logical forms, which can lead to the increase of the number of denotations. That needs much more time comparing to interpret terms from logical forms only. We use dynamic programming on denotations to recover the full set, following the work of [13, 22]. A necessary condition for dynamic programming to work is denotationally invariant semantic function g , such that the denotation of the resulting logical form $g(s_1, s_2)$ only depends on the denotations of s_1 and s_2 . In the arithmetic domain, the result of an equation can be computed recursively and independently which certainly satisfies this requirement.

Our primary purpose is to collapse logical forms with the same denotation together, so that given a denotation d the candidate set Ω (Fig. 1(a)) can be returned directly. In order to speed up the lookup efficiency, we store the pair (d, Ω) in advance.

Table 1. The base case set with several pairs of $\langle u, s \rangle$

Utterance	Logical form	Denotation
One plus two	$\langle \text{Go} [1 + 2] \text{End} \rangle$	3.0
Three minus four times five	$\langle \text{Go} [3 - (4 * 5)] \text{End} \rangle$	-17.0
One times two divide three	$\langle \text{Go} ((1 * 2) / 3) \text{End} \rangle$	0.667
Two divide four plus five	$\langle \text{Go} [(2 / 4) + 5] \text{End} \rangle$	5.5
Five divide one times two plus three	$\langle \text{Go} [((5 / 1) * 2) + 3] \text{End} \rangle$	13.0
Four minus two times three plus one	$\langle \text{Go} [[4 - (2 * 3)] + 1] \text{End} \rangle$	-1.0
Three divide four minus five plus two	$\langle \text{Go} [[(3 / 4) - 5] + 2] \text{End} \rangle$	-2.25

3.2 Filter Candidate Logical Forms

In order to filter out incorrect logical forms, we utilize the fact that similar utterances should have similar logical forms to reduce the searching space. We build up a base case set B (Table 1) that stores several $\langle u_b, s_b \rangle$ pairs with varying utterance length. Specifically, given an input pair $\langle u_i, d_i \rangle$, we iterate the base case set to find the one which shares the most similarity with input utterance u_i .

$$\tilde{u}_b = \arg \max_{u_b \in B} e(\phi(u_b), \phi(u_i)), \quad (11)$$

where ϕ extracts features from utterance u_b and u_i , here we use bag-of-words as feature extraction function, and we simply counts the shared features as the feature similarity measurement function e .

After finding the base case utterance \tilde{u}_b which is closest to the input utterance u_i , we use the corresponding base case logical form \tilde{s}_b to filter the candidate set Ω .

$$\Gamma = \{s | e(\phi(\tilde{u}_b), \phi(u_i)) = e(\phi(\tilde{s}_b), \phi(s)), s \in \Omega\} \quad (12)$$

We use a new set Γ (Fig. 1(b)) to store these updated candidate logical forms that have similar features with the base case $\langle \tilde{u}_b, \tilde{s}_b \rangle$. For example, an input utterance ‘‘one plus three’’ have denotation 4, which can be used to construct the candidate set Ω . After iterating the base case set, we find the best base case utterance ‘‘one plus two’’ is the closet one to input utterance, and its corresponding logical form is $\tilde{s}_b = \langle Go[1 + 2]End \rangle$. By iterating the candidate set Ω , logical form $\langle Go[1 + 3]End \rangle$ is remained in Γ , but $\langle Go[5 - 1]End \rangle$ is eliminated.

To further determine the most probable one from set Γ , we use the Seq2Seq model to examine the loss of each candidate and select the one with least loss to return for training.

$$\tilde{s}_i = \arg \max_{s \in \Gamma} p_{Seq2Seq}(s) \quad (13)$$

The selected logical form \tilde{s}_i is paired with its utterance to form a training example $\langle u_i, \tilde{s}_i \rangle$ which feeds back to the Seq2Seq model for training (Fig. 1 (c)–(d)).

4 Experimental Studies

4.1 Dataset

In our experimental studies, we randomly generate 8000 utterances with varying length from 3 to 7, split into a training set of 6000 training examples and 2000 test examples. Each utterance consists of integers from one to five and four operators ‘plus’, ‘minus’, ‘times’, ‘divide’. Every utterance represents a legal arithmetic expression. Even the scope is quite limited, the searching space for an equation with length equal to seven is still numerous: $5^4 * 4^3 = 4 * 10^4$, which is caused by the rich compositionality of arithmetic equations. Besides, this nature gives rise to one denotation can correspond to much more logical forms compared with other domains, resulting in increasing noises for inference.

We select two ways to represent logical forms, both of them are represented by Arabic numerals and executable operators, but one is inserted with brackets to denote the calculation order, linearized from tree structure, the other assumes knowing calculation order without brackets for denotation. The base case set consists of seven samples, the brackets are omitted directly when considering logical forms with calculation knowledge (Table 1).

The involvement of brackets largely increases the meaning of our logical form, for the reason that it actually represents tree structure with logic reasoning. Besides, not only has the model to learn soft alignment for brackets which are not introduced in the utterance explicitly, but it has to learn the brackets matching relationship.

For the convenience of data preprocessing and vectorization, we manually set a maximum length for the input and output sentence, where the blank position will be automatically filled by ‘PAD’ mark. We construct 3 layers of LSTM on both encoder and decoder side with 20 hidden units for each layer. An embedding and a softmax layer is inserted as the first and last layer. Dropout is used for regularizing the model with a constant rate 0.3. Dropout operators are used between different LSTM layers and for the hidden layers before the softmax classifier. This technique can significantly reduce overfitting, especially on datasets of small size. Dimensions of hidden vector and word embedding are set to 20.

We use the RMSProp algorithm to update parameters with learning rate 0.001 and smoothing constant 0.9. Parameters are randomly initialized from a uniform distribution $\mathcal{U}(-0.05, 0.05)$. We run the model for 200 epochs.

4.2 Results and Analysis

We report the results with the Seq2Seq model on two variants, i.e., with brackets noting calculation order and without brackets as logical forms. To compare the performance of weak supervision, we also test the performance of traditional training with gold standard logical form. The result is reported on the accuracy of denotation – the portion of input sentences are translated into the correct denotation. Table 2 presents the comparison.

Table 2. Overall accuracy on denotation

Method	Seq2Seq	
	Without brackets	With brackets
Train with logical form	100.0%	92.4%
Train with denotation	72.7%	69.2%

Overall, the accuracy with logical forms as supervision is much higher than with denotations, for the reason that training by gold logical forms does not bring any noises. On the other hand, the spurious logical forms affect our model’s performance unavoidably. For instance, an utterance “Five plus three times four” can be mistakenly translated into $[5 + (4 * 3)]$, which has the correct denotation. This is reasonable because we adopt bag-of-words to measure similarity and this method does not take the order into consideration. Besides, not all of the training logical forms returned by our inference process are correct, but at least they have the correct denotation. So it is noticeable that even only approximately 45% logical forms (Fig. 2) returned for training are correct, the accuracy on denotation of our model is much higher than this limit.

To find the best logical form for training, the final pick decision is made by the ongoing training Seq2Seq model, this training-by-prediction policy makes the model difficult to converge (Fig. 2). But we can see this policy is effective

to correct its previous prediction, which we call the ability of self-correction. This proves Seq2Seq model can learn word meanings and compositionality with noises.

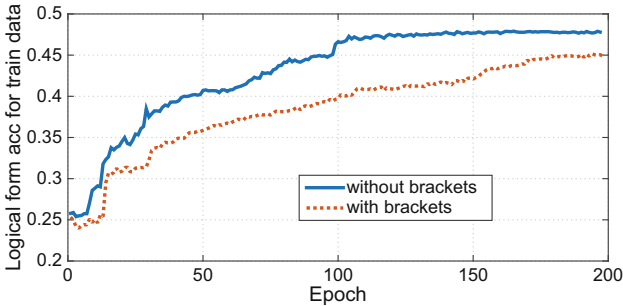


Fig. 2. The percentage of correct logical form returned for training as the number of epoch increases.

In addition, the performance of training on logical forms with brackets is inferior to the model without brackets, which makes sense that longer sequence add difficulty for learning and brackets introduce more complicated mapping relationships. However, by utilizing soft alignment, the model can learn tree structure successfully with only denotation as supervision.

5 Conclusion

In this paper, we present an encoder-decoder neural network model for mapping natural language to their meaning representations with only denotation as supervision. We use dynamic programming to infer logical forms from denotations, and utilize similarity measurement to reduce the searching space. Also, curriculum learning strategy is adopted to smooth and accelerate the learning process. Under the policy training-by-predictions, our model has the ability of self-correction. We apply our model to the arithmetic domain and experimental results show that this model can somehow learn compositionality of terms and even semantics. One major problem remained in our work is that the model has limited applications. Although the example we test is rather simple, the expansibility to other fields and application scenarios is promising due to the few hand-engineered features and its capability of learning structured form. It would be interesting to learn a question-answering model with only question-answer pairs, or apply it to communicating with robots - for example, we are able to give orders in natural language to robots.

Acknowledgement. This work is supported by the National Science Foundation of China Nos. 61401012 and 61305047.

References

1. Artzi, Y., Zettlemoyer, L.: Weakly supervised learning of semantic parsers for mapping instructions to actions. *Trans. Assoc. Comput. Linguist.* **1**, 49–62 (2013)
2. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate (2014). arXiv preprint [arXiv:1409.0473](https://arxiv.org/abs/1409.0473)
3. Cho, K., Merriënboer, B.V., Bahdanau, D., Bengio, Y.: On the properties of neural machine translation: encoder-decoder approaches (2014). arXiv preprint [arXiv:1409.1259](https://arxiv.org/abs/1409.1259)
4. Cho, K., Merriënboer, B.V., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. *EMNLP*, pp. 1724–1734 (2014)
5. Clarke, J., Goldwasser, D., Chang, M.W., Roth, D.: Driving semantic parsing from the world’s response. In: Fourteenth Conference on Computational Natural Language Learning, pp. 18–27 (2010)
6. Dong, L., Lapata, M.: Language to logical form with neural attention. In: Meeting of the Association for Computational Linguistics, pp. 33–43 (2016)
7. Ge, R., Mooney, R.J.: A statistical semantic parser that integrates syntax and semantics. In: Conference on Computational Natural Language Learning, pp. 9–16 (2005)
8. Graves, A., Wayne, G., Danihelka, I.: Neural Turing machines (2014). arXiv preprint [arXiv:1410.5401](https://arxiv.org/abs/1410.5401)
9. Hermann, K.M., Kočiský, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., Blunsom, P.: Teaching machines to read and comprehend. In: NIPS, pp. 1693–1701 (2015)
10. Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural Comput.* **9**(8), 1735 (1997)
11. Krishnamurthy, J., Mitchell, T.M.: Weakly supervised training of semantic parsers. In: Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pp. 754–765 (2012)
12. Lao, N., Mitchell, T., Cohen, W.W.: Random walk inference and learning in a large scale knowledge base. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, pp. 529–539 (2011)
13. Lau, T., Wolfman, S.A., Domingos, P., Weld, D.S.: Programming by demonstration using version space algebra. *Mach. Learn.* **53**(1), 111–156 (2003)
14. Liang, C., Berant, J., Le, Q., Forbus, K.D., Lao, N.: Neural symbolic machines: Learning semantic parsers on freebase with weak supervision (2016). arXiv preprint [arXiv:1611.00020](https://arxiv.org/abs/1611.00020)
15. Liang, P., Jordan, M.I., Klein, D.: Learning dependency-based compositional semantics. In: Meeting of the Association for Computational Linguistics: Human Language Technologies, pp. 590–599 (2011)
16. Liang, P., Potts, C.: Bringing machine learning and compositional semantics together. *Annu. Rev. Linguist.* **1**(1), 355–376 (2015)
17. Lu, W., Ng, H.T., Lee, W.S., Zettlemoyer, L.S.: A generative model for parsing natural language to meaning representations. In: *EMNLP*, pp. 783–792 (2008)
18. Miller, S., Stallard, D., Bobrow, R., Schwartz, R.: A fully statistical approach to natural language interfaces. In: Proceedings of the 34th annual meeting on Association for Computational Linguistics, pp. 55–61 (2002)

19. Neelakantan, A., Le, Q.V., Abadi, M., Mccallum, A., Amodei, D.: Learning a natural language interface with neural programmer (2016). arXiv preprint [arXiv:1611.08945](https://arxiv.org/abs/1611.08945)
20. Neelakantan, A., Le, Q.V., Sutskever, I.: Neural programmer: inducing latent programs with gradient descent. *Computer Science* (2015)
21. Pasupat, P., Liang, P.: Compositional semantic parsing on semi-structured tables (2015). arXiv preprint [arXiv:1508.00305](https://arxiv.org/abs/1508.00305)
22. Pasupat, P., Liang, P.: Inferring logical forms from denotations (2016). arXiv preprint [arXiv:1606.06900](https://arxiv.org/abs/1606.06900)
23. Reddy, S., Lapata, M., Steedman, M.: Large-scale semantic parsing without question-answer pairs. *Trans. Assoc. Comput. Linguist.* **2**, 377–392 (2014)
24. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. *Adv. Neural Inf. Proc. Syst.* **4**, 3104–3112 (2014)
25. Vinyals, O., Kaiser, L., Koo, T., Petrov, S., Sutskever, I., Hinton, G.: Grammar as a foreign language. eprint Arxiv (2014)
26. Wang, Y., Berant, J., Liang, P.: Building a semantic parser overnight. In: Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing, pp. 1332–1342 (2015)
27. Zettlemoyer, L.: Learning to map sentences to logical form. eprint Arxiv, pp. 658–666 (2010)



Phrase-Level Grouping for Lexical Gap Resolution in Korean-Vietnamese SMT

Seung Woo Cho^(✉) , Eui-Hyeon Lee, and Jong-Hyeok Lee

Department of Computer Science and Engineering,
Pohang University of Science and Technology, Pohang, Republic of Korea
{itswc, jekyllbox, jhlee}@postech.ac.kr

Abstract. A lexical gap easily leads to word alignment errors, which impairs a translation quality. This paper proposes some simple ideas to resolve the difficulty of handling the lexical gap. In morphologically rich languages, a predicate has a complex structure consisting of many morphemes, so we mainly address the issue of how to group the component morphemes by employing morpho-syntactic filters and statistical information from the SMT phrase table. In addition, we abstract grouping results depending on a lexical choice of the target side to enhance translation probabilities. In the experiment, we not only investigate how each method has an effect on Korean-to-Vietnamese SMT, but also show a promising improvement of BLEU score.

Keywords: Statistical machine translation · Lexical gap resolution
Morpheme group · Multi-word expression
Korean-Vietnamese translation

1 Introduction

A multi-word expression (MWE) is a special lexical unit, which consists of several words. Since the meaning of the MWE has never been predicted from combinations of meanings of words in the MWE, the MWE should be regarded as a unique token to express the original meaning of the MWE [1]. With respect to the lexical gap [2] in machine translation of the distant language pair, a single MWE in a source side can be translated into a single word or morpheme in a target side.

Phrases in the phrase-based statistical machine translation (PBSMT) [10] systems are constructed based on the result of word alignments without sophisticated linguistic knowledge. In PBSMT, words that belong to the MWE of the source side are respectively aligned to the target side due to word-level alignments. This drawback can pose to a low quality of word alignments. Further, even if MWEs are constructed as a single phrase in the phrase table, the translation probability of the phrase pairs is not optimized due to the low quality of word alignments. Consequently, the insufficient processing of MWEs impairs the translation quality.

In this study, we analyze an experimental study of Korean-Vietnamese PBSMT and prove that our methods take effect on the lexical gap problem. Before we propose our methods, we introduce the morphological features on both of languages.

Korean is one of agglutinative languages, which has rich derivational and inflectional morphology [13]. A stem and affixes have to be separated by the morphological analysis to prevent the frequency imbalance in translations of Korean words.

On the other hand, Vietnamese belongs to the isolated language, which is a morphologically poor language. To construct the grammatical relationship and meaning, the morphological analysis should be performed so that monosyllables are combined to each other based on the word order and grammatical roles of functional words [22].

We sample an example from training data to show what issues can occur. Figure 1 describes an ideal alignment of the sample sentence after the morphological analysis. Each number indicates ideal alignments of morphemes. In Korean, brackets show Korean word boundaries. Also, all the morphemes in the second line of Korean belong to a predicate. We find out three interesting points from Fig. 1.

First, some case markers in Korean such as ‘neun’ and ‘leul’ are not aligned with any morpheme in Vietnamese grammatically. Second, a translation ambiguity on homonyms can occur. In case of the Korean morpheme ‘geu’, morphemes corresponding in Vietnamese are different. Last of all, for example of the lexical gap, several morphemes ‘l’, ‘su’, ‘iss’ and ‘da’ in the Korean predicate should be grouped and be aligned with a single morpheme ‘cô_thể’, in Vietnamese. These morphological differences existing in the Korean-Vietnamese language pair can induce alignment errors.

In this paper, we group morphemes of a predicate as the unique token by elaborating morpho-syntactic filters and statistical information of the phrase table in Moses [9]. We mainly focus on extracting phrase pairs in which N-to-1 alignments appear. Further, we abstract morpheme groups depending on the lexical choice of the target side to eliminate the data sparseness triggered from grouping morphemes.

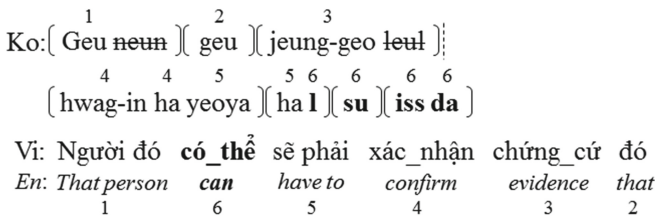


Fig. 1. The alignment example of the morpheme between Korean and Vietnamese

We introduce previous researches in Sect. 2. Our methods are described in Sect. 3. Experimental settings are addressed in Sect. 4. Experimental results are presented in Sect. 5. We conclude this paper in Sect. 6.

2 Related Work

To translate MWEs properly, various research has been conducted. In [18], Sakata et al. employ the large-size dictionary which provides a wide coverage by considering word, phrase and clause-level translation patterns. Similar to [18], Todiraşcu and Navlea identify syntactic collocations in which the noun is following the verb based on the bilingual MWE (BMWE) dictionary [21]. The dictionary-based approach provides useful context to detect MWEs, but the cost to build the dictionaries is not affordable for new language pairs.

Extracting BMWEs by using a monolingual corpus is proposed by [3, 17, 20, 21]. In [21], they use the log likelihood [5] to extract candidates of MWEs for each language independently. Also, linguistic information such as the part of speech (PoS) and the lemma form of the word is employed to eliminate irrelevant candidates. However, since their approach does not consider alignments between MWEs, the information loss can be posed.

In Chinese-English translation, Ren et al. focus on extracting BMWEs in technical domains. They extract MWEs based on the log likelihood ratio from the Chinese text of a parallel corpus at first [17]. They select all translations of Chinese MWEs from the phrase table as candidates of English MWEs. All candidates are classified by a perceptron-based classifier.

Bouamor et al. produce patterns of MWEs by linguistic filters on the language pair [3]. With these patterns, they extract all candidate pairs of BMWEs from the parallel sentences. The jaccard index is used to classify candidate pairs as BMWEs. However, annotated data, sophisticated linguistic patterns and detection tools [20] are still too expensive to apply for new language pairs.

In [11], the relative frequency and co-occurrence values calculated from the result of word alignments are employed to decide whether the bilingual phrase pair can belong to BMWEs or not. With this statistical measurement, they construct linguistic criteria by using morpho-syntactic filters to increase the precision of extracting BMWEs. However, even if a number of BMWEs are applied, no improvement on BLEU scores is demonstrated.

They report two kinds of errors. First, determiners and particles in BMWEs are inserted at the wrong place in the translation hypothesis. The coverage of their method is too wide to obtain patterns in high quality. Second, the data sparseness from encapsulations of BMWEs poses to loss of statistics. This drawback impairs translating inflections of words in the n-gram models including back-off models.

Since their approach extracts BMWEs from the phrase table without additional dictionaries and detection tools, we adopt the grouping technique in [11] to extract BMWEs on the morpheme representation. By adjusting the number

of words in the target side, we can obtain the phrase pairs that belong to the lexical gap easily. However, the patterns in the low quality and the data sparseness should be eliminated to improve translation quality.

3 Methodology

In the report of [11], the main problems are the low quality of patterns and the data sparseness. In this section, we mainly address how to resolve these drawbacks through our grouping and abstraction methods.

3.1 Grouping Morphemes of a Predicate

In this experiment, we mainly focused on extracting meaningful morpheme groups based on the frequency information from the phrase table. Since over-generations of morpheme groups can lead to the low quality of patterns easily, we targeted phrase pairs that belong to predicates, not for entire phrase pairs.

The statistical information we employed is as follows.

$$c(e, f) = c(e) \times p(f|e) \quad (1)$$

$c(e)$ indicates a frequency of a target phrase in the phrase table. And $p(f|e)$ means a probability of a source phrase given the target phrase. $c(e, f)$ is a co-occurrence value of the source phrase and the target phrase in the phrase table.

We hypothesized that this values guide us to group morphemes more accurately than directly applying translation probabilities which contains the error propagated from word alignments. In a huge volume of the parallel corpus, $c(e, f)$ can be used to indicate how well the source phrase and the target phrase are aligned intuitively.

We normalized $c(e, f)$ to extract important phrase pairs among entire phrase pairs based on a z-score formula as follows.

$$z = \frac{c(e, f) - \mu_{c(e, f)}}{\sigma_{c(e, f)}} \quad (2)$$

We excluded useless phrase pairs in which z of the phrase pair is below a specific threshold or phrase pairs do not belong to the criterion defined by morpho-syntactic filters as follows.

First, at least, the source phrase should have one verb or adjective morpheme. Second, the source phrase should not have pronoun, proper noun, numeral, special marks and any case marker which never appears in the predicate. Lastly, the target phrase should have only one verb or adjective morpheme.

We extracted 69 phrase pairs from the phrase table in the baseline system. The morphemes in the source side were grouped by a source phrase pattern of each extracted phrase pair. We retrained our PBSMT system to apply changes of the training corpus.

In Table 1, expressions on the left side of the ‘/’ mark are the surface form of each morpheme, the other side means PoS information. With regards to a representation of each group, we connected all surface forms of each morpheme with ‘_’ mark. Also, PoS information of each morpheme was linked with ‘+’ mark.

3.2 Abstracting Morpheme Groups

After grouping morphemes, the data sparseness can occur [11]. Therefore, we abstracted morpheme groups to overcome the drawback in grouping morphemes.

Among extracted phrase pairs, we found out that several source phrases have the same lexical choice on the target side as shown in Table 2. We abstracted source phrases depending on the lexical choice of Vietnamese phrases to prevent the data sparseness which was reported in [11].

Depending on the lexical choice of the target side, 69 patterns were abstracted into 33 sets. And then, we abstracted morpheme groups in the parallel corpus and retrain our PBSMT system.

Table 1. The examples of grouping morphemes in the source side

Morpheme representation	Grouping representation
l/fmotg su/CMD iss/YBH	l_su_iss/fmotg+CMD+YBH
ji/fmoc anh/YA	ji_anh/fmoc+YA

Table 2. The examples of abstracting source phrases depending on the lexical choices of the target side

Source phrase	Target phrase
l_su_iss/fmotg+CMD+YBH	cô_thể/Aa
l_su_iss/fmotg+CMD+YBD	
eul_su_iss/fmotg+CMD+YBH	
ha_l_su_iss/fpd+fmotg+CMD+YBD	
su_eobs/CMD+YBH	không/R
ji_anh/fmoc+YA	
ji_mosha/fmoc+YBD	
ji_mosha/fmoc+YA	
ji_anh_go/fmoc+YA+fmoc	
ha_ji_anh/fpd+fmoc+YA	

In contrast to the grouping experiment, the surface form of each abstraction was replaced with the distinct mark of each abstraction. And PoS information of each abstraction was unified as a designated mark ‘SYN’.

4 Experiments

In this section, we explain our baseline system and what data we used. Also, we describe how we evaluated proposed methods.

4.1 Parallel Data

1.4M parallel sentences were used to build our PBSMT system. We divided the parallel corpus into three corpora for training, tuning and testing. The statistics of each experimental corpus are shown in Table 3.

We find out that the average number of morpheme between Korean and Vietnamese quite differs. This difference shows that the lexical gap may frequently appear from Korean to Vietnamese.

Table 3. The statistics of each experimental corpus

Corpus	Sentence	Ko. morpheme	Vi. morpheme
Training	1,475,654	20,930,173	17,091,843
Tuning	5,000	82,651	69,237
Test	5,000	84,291	70,660

4.2 Baseline System

As our baseline system, we used the Moses toolkit and employed a KEN language modeling toolkit [7]. A GIZA++ toolkit [15] was also used for word alignments. We constructed a 3-gram language model using Vietnamese sentences in the training corpus.

With regards to tuning the PBSMT system, MERT [15] was employed. We added an unrestricted distortion parameter for long distance movements while we decode the test sentences.

In Korean, the word order is Subject-Object-Verb dominant and highly scrambled [19]. In contrast to Korean, Vietnamese follows a Subject-Verb-Object word order. We expect that the difference of the word order between two languages is minimized through this option [6].

Since we would like to evaluate the methods for lexical gap resolution, we eliminated the other morphological differences before the experiments. First, for the morphological analysis, we employed an in-house PoS tagger [16] on Korean and an external PoS tagger [4, 14] on Vietnamese. We applied PoS information of each morpheme [12] for disambiguation in translations of homonyms. Second, we deleted nominative case markers, accusative case markers and topic case markers, which do not correspond with any morpheme in the other side [6, 12].

4.3 Evaluation

We evaluated our system by BLEU scores computed on the monosyllable representation, in that original Vietnamese sentence consists of monosyllables. Further, we follow [8] to prove a statistical significance of our system with p-value 0.05.

5 Results and Discussion

We addressed two preprocessing techniques to resolve the lexical gap in the distant language pair. We applied our preprocessing steps into the baseline system in a sequence.

BLEU scores for all experiments are shown in Table 4. We present the case of adding the unrestricted distortion parameter as a modified parameter in Table 4. P-values from most of the experiments are computed as under 0.05 except the abstraction experiment with the default parameter ($p = 0.111$).

We confirm that all experiments improve translation qualities. First of all, the unrestricted distortion option improves the translation quality for all cases dramatically. The long distance movements mitigate the difference of the word order in the distant language pair efficiently. The grouping technique raises BLEU scores of 0.7 points, compared to the result of the baseline system with the modified parameter. Also, abstracting morpheme groups improves BLEU scores of 0.23 points. As a result of all preprocessing steps, compared to our baseline system, BLEU scores of 0.93 points increase.

To investigate the reason of improvements, we select 3 abstractions based on the frequency of each abstraction in the test corpus. Table 5 shows the number of occurrences of top 5 abstractions in the test corpus. We extract the most frequent

Table 4. BLEU scores for each experiment

System	Default parameter	Modified parameter
Baseline	35.19 (0.0%)	37.42 (6.3%)
+Grouping	35.52 (0.9%)	38.12 (8.3%)
+Abstraction	35.62 (1.2%)	38.35 (9.0%)

Table 5. The frequency of top 5 abstractions in the test corpus

Index	Target phrase	Frequency
7	không/R	316
5	cô_thê ² /Aa	289
2	bị/Vv	183
4	đang/R	172
14	Phái/Vv	170

Table 6. Phrase scores of the most frequent phrase pair in top 3 abstractions

(a) Phrase scores in the baseline system						
Index	Source	Target	$\varphi(f e)$	$lex(f e)$	$\varphi(e f)$	$lex(e f)$
7	ji/fmoc anh/YA	không/R	0.12074	0.00643	0.02192	0.02607
5	ha/fpd l/fmotg su/CMD iss/YBH	cô_thế/Aa	0.01936	0.00063	0.29677	0.31975
2	eo/fmoc ji/YA	bi/Vv	0.01809	0.001	0.0408	0.0236

(b) Phrase scores in the grouping experiment						
Index	Source	Target	$\varphi(f e)$	$lex(f e)$	$\varphi(e f)$	$lex(e f)$
7	ji_anh/fmoc+YA	không/R	0.11005	0.05809	0.01917	0.02715
5	ha_l_su_iss/ fpd+fmotg+CMD+YBH	cô_thế/Aa	0.14048	0.13688	0.56849	0.61306
2	eo_ji/fmoc+YA	bi/Vv	0.03344	0.02504	0.05557	0.04732

(c) Phrase scores in the abstraction experiment						
Index	Source	Target	$\varphi(f e)$	$lex(f e)$	$\varphi(e f)$	$lex(e f)$
7	Abstract7/SYN	không/R	0.17389	0.08667	0.02029	0.02745
5	Abstract5/SYN	cô_thế/Aa	0.51278	0.49209	0.56618	0.60473
2	Abstract2/SYN	bi/Vv	0.07608	0.05196	0.0654	0.05911

phrase pair of each selected abstraction and analyze the statistical changes of extracted phrase pairs in the phrase table.

The scores expressed in Table 6 are important parameters which decide what the quality of translation hypotheses are generated in the decoder. $\varphi(f|e)$ is an inverse phrase translation probability, which means the probability of the Korean phrase (f) given the Vietnamese phrase (e). $lex(f|e)$ indicates an inverse lexical weighting, which indicates how reliable the Korean phrase (f) matches up given the Vietnamese phrase (e). $\varphi(e|f)$ and $lex(e|f)$ are the scores of the symmetric situation of $\varphi(f|e)$ and $lex(f|e)$.

With respect to phrase scores, our techniques succeed to increase all the phrase scores and directly contribute to improve translation hypotheses, compared to the baseline system. Morphemes that belong to the lexical gap in the morphologically rich language are successfully grouped together through the statistical approach. However, even if our grouping technique has positive effects on phrase scores, the data sparseness still exists due to the creation of morpheme groups.

In Table 6, we find out that our abstraction technique helps to overcome the drawback of the grouping technique. The abstraction technique reduces ambiguities which occur from the target side to the source side. Increments of all indirect phrase scores reflect this merit. Even if the number of abstractions is

not changed significantly, we confirm that the abstraction technique provides sufficient aids to reduce the data sparseness by abstracting morpheme groups.

6 Conclusion

We showed the lexical gap in Korean-Vietnamese translation and resolved the lexical gap through simple preprocessing steps.

In particular, we concentrated on grouping morphemes of the complex predicate. Based on successful results of the grouping technique, we also presented the abstraction technique, which prevents the data sparseness. However, our methods were limited on predicates of the morphologically rich language.

In the future, we will study on a machine learning algorithm for the grouping technique which has a wider coverage. Also, on the basis of the high similarity of morphological structures, we will experiment with reducing differences of syntactic structures like as preordering techniques.

Acknowledgment. This work was partly supported by the ICT R&D program of MSIP/IITP [R7119-16-1001, Core technology development of the real-time simultaneous speech translation based on knowledge enhancement], the ICT Consilience Creative Program of MSIP/IITP [R0346-16-1007] and SYSTRAN.

References

1. Baldwin, T., Kim, S.N.: Multiword expressions. In: Handbook of Natural Language Processing, 2nd edn., pp. 267–292. Chapman and Hall/CRC (2010)
2. Bentivogli, L., Pianta, E.: Looking for lexical gaps. In: Proceedings of the ninth EURALEX International Congress, pp. 8–12. Universität Stuttgart, Stuttgart (2000)
3. Bouamor, D., Semmar, N., Zweigenbaum, P.: A study in using English-Arabic multi-word expressions for statistical machine translation. In: 4th International Conference on Arabic Language Processing (2012)
4. Dien, D., Thuy, V.: A maximum entropy approach for vietnamese word segmentation. In: Proceedings of 4th IEEE International Conference on Computer Science Research, Innovation and Vision of the Future 2006 (RIVF 06), pp. 12–16 (2006)
5. Dunning, T.: Accurate methods for the statistics of surprise and coincidence. *Comput. Linguist.* **19**(1), 61–74 (1993)
6. El-Kahlout, I.D., Ofazer, K.: Exploiting morphology and local word reordering in English-to-Turkish phrase-based statistical machine translation. *IEEE Trans. Audio Speech Lang. Process.* **18**(6), 1313–1322 (2010)
7. Heafield, K.: KenLM: faster and smaller language model queries. In: Proceedings of the Sixth Workshop on Statistical Machine Translation, pp. 187–197. Association for Computational Linguistics (2011)
8. Koehn, P.: Statistical significance tests for machine translation evaluation. In: *EMNLP*, pp. 388–395. Citeseer (2004)

9. Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., et al.: Moses: open source toolkit for statistical machine translation. In: Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions, pp. 177–180. Association for Computational Linguistics (2007)
10. Koehn, P., Och, F.J., Marcu, D.: Statistical phrase-based translation. In: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1, pp. 48–54. Association for Computational Linguistics (2003)
11. Lambert, P., Banchs, R.: Grouping multi-word expressions according to part-of-speech in statistical machine translation. Multi-word-expressions in a multilingual context, p. 9 (2006)
12. Lee, J., Lee, D., Lee, G.G.: Improving phrase-based Korean-English statistical machine translation. In: INTERSPEECH (2006)
13. Li, S., Wong, D.F., Chao, L.S.: Korean-Chinese statistical translation model. In: 2012 International Conference on Machine Learning and Cybernetics (ICMLC), vol. 2, pp. 767–772. IEEE (2012)
14. Nghiem, M., Dinh, D., Nguyen, M.: Improving Vietnamese pos tagging by integrating a rich feature set and Support Vector Machines. In: IEEE International Conference on Research, Innovation and Vision for the Future, RIVF 2008, pp. 128–133. IEEE (2008)
15. Och, F.J., Ney, H.: A systematic comparison of various statistical alignment models. *Comput. Linguist.* **29**(1), 19–51 (2003)
16. Oh-Woog, K., Yujin, C., Mi-Young, K., Dong-Won, R., Moon-Ki, L., Jong-Hyeok, L.: Korean morphological analyzer and part-of-speech tagger based on cyb algorithm using syllable information. In: Proceedings of the 11th Annual Conference on Human and Cognitive Language Technology, pp. 76–87 (1999)
17. Ren, Z., Lü, Y., Cao, J., Liu, Q., Huang, Y.: Improving statistical machine translation using domain bilingual multiword expressions. In: Proceedings of the Workshop on Multiword Expressions: Identification, Interpretation, Disambiguation and Applications, pp. 47–54. Association for Computational Linguistics (2009)
18. Sakata, J., Tokuhisa, M., Murata, M., et al.: Machine translation method based on non-compositional semantics (word-level sentence-pattern-based MT). In: Hasida, K., Purwarianti, A. (eds.) International Conference of the Pacific Association for Computational Linguistics, pp. 225–237. Springer, Singapore (2015). https://doi.org/10.1007/978-981-10-0515-2_16
19. Shin, S.: Corpus-based study of word order variations in Korean. In: Proceedings of the Corpus Linguistics Conference (CL 2007), pp. 27–30. Citeseer (2007)
20. Skadina, I., Rozis, R.: Multi-word expressions in English-Latvian. In: Human Language Technologies-The Baltic Perspective: Proceedings of the Seventh International Conference Baltic HLT 2016, vol. 289, p. 97. IOS Press (2016)
21. Todiraşcu, A., Navlea, M.: Aligning verb+ noun collocations to improve a French-Romanian FSMT system. In: Multi-word units in Machine Translation and Translation Technologies, MUMTTT 2015, p. 37 (2015)
22. Tran, P., Dinh, D., Nguyen, L.H.: Word re-segmentation in chinese-vietnamese machine translation. *ACM Trans. Asian Low-Res. Lang. Inf. Process. (TALLIP)* **16**(2), 12 (2016)



Enhancing Pivot Translation Using Grammatical and Morphological Information

Hai-Long Trieu^(✉) and Le-Minh Nguyen

School of Information Science,
Japan Advanced Institute of Science and Technology, Nomi, Japan
{trieulh,nguyenml}@jaist.ac.jp

Abstract. Pivot translation can be one of the solutions to overcome the problem of unavailable large bilingual corpora for training statistical machine translation models. Nevertheless, the conventional pivot method, which connect source to target phrases via common pivot phrases, lacks some potential connections when pivoting via the surface form of pivot phrases. In this work, we improve the pivot translation method by integrating grammatical and morphological information to connect pivot phrases instead of using only the surface form. Experiments were conducted on several Southeast Asian low-resource language pairs: Indonesian-Vietnamese, Malay-Vietnamese, and Filipino-Vietnamese. By integrating grammatical and morphological information, the proposed method achieved a significant improvement of 0.5 BLEU points. This showed the effectiveness of integrating grammatical and morphological features to pivot translation.

Keywords: Statistical machine translation · Pivot methods
Factored models · Part-of-speech tags · Lemma forms

1 Introduction

Statistical machine translation (SMT) models require large bilingual corpora to learn reliable translation probabilities [10, 24]. Nevertheless, such large bilingual corpora are unavailable for most language pairs. In order to overcome the problem, some methods have been proposed such as: building more bilingual corpora [3, 8, 21], using monolingual data [15, 18, 19], and pivot translation via intermediate languages [4, 5, 23, 25].

Pivot methods can be one of the solution for SMT when large bilingual corpora are unavailable. In the pivot methods, two bilingual corpora of source-pivot and pivot-target language pairs are used to generate translations of the source-target language pair via the pivot language(s). The triangulation approach [4, 23, 25] is one of the best approaches in pivot methods, in which source and target phrases are connected via common pivot phrases in the source-pivot and pivot-target phrase tables trained on the source-pivot and pivot-target bilingual corpora. Previous work of the triangulation approach based on the surface

form of pivot phrases to connect source and target phrases, which does not take advantage full potential connections of the phrase tables via pivot phrases. This work aims to improve the triangulation approach by integrating linguistic information in terms of grammatical (part-of-speech) and morphological (lemma) of pivot phrases to enrich knowledge for pivot translation.

In this work, we propose a method to connect pivot phrases based on grammatical and morphological information. Instead of connecting based on the surface form of pivot phrases as in the conventional triangulation method, grammatical and morphological information were integrated to extract more informative connections between pivot phrases. Specifically, the part-of-speech tags and lemma forms of words in the pivot sides of the source-pivot and pivot-target bilingual corpora were generated to enrich knowledge for pivoting via pivot phrases. Then, the bilingual corpora were used to train factored models, in which pivot phrases contain grammatical and morphological information instead of only the surface form. Experiments were conducted on several Southeast Asian low-resource language pairs: Indonesian-Vietnamese, Malay-Vietnamese, and Filipino-Vietnamese. Experimental results show that our method outperforms the conventional triangulation method with 0.5 BLEU points. Statistical significance test were carefully conducted to verify the improvement. The experiments showed the effectiveness of grammatical and morphological features in pivot translation. The corpora and source code used in this research are available at the repository.¹

2 Related Work

There are three main approaches of pivot translation: cascade, synthetic, and phrase pivot translation (or triangulation). In the first approach, cascade, sentences are translated from source language to pivot then to target language [5, 23]. In the second approach, synthetic [5], a pseudo source-target bilingual corpora is generated using source-pivot and pivot-target bilingual corpora. For instance, the pivot side of the source-pivot bilingual corpus can be translated into the target language using the pivot-target translation model. In the third approach, triangulation [4, 23, 25], source-pivot and pivot-target phrase tables are trained from the source-pivot and pivot-target bilingual corpora. Then, source and target phrases are connected based on common pivot phrases. The triangulation approach has shown to be the most effective in pivot methods [6, 23].

Some previous work has been proposed to improve the conventional triangulation approach. Zhu et al., [26] employed more connections of source and target phrases using based on a Markov random walks model. El Kholy et al., [6] based on word alignment information as strength features to filter low quality phrase pairs. Zhu et al., [27] proposed a method for pivoting via co-occurrence counts of phrase pairs to improve phrase translation’s scores estimated by the triangulation. In this work, we used the part-of-speech information and lemma forms of pivot phrases to improve the triangulation based on only the surface

¹ <https://github.com/nguyenlab/pivot-linguistics>.

form of pivot phrases. In order to add the linguistic information, we based on the factored training [12] to integrate the part-of-speech and lemma information, which has shown to be effective in adding linguistic knowledge for training SMT models.

3 Approaches

In this work, we proposed using grammatical and morphological knowledge to improve the conventional triangulation method. We recall the triangulation method before we present integrating grammatical and morphological information.

3.1 Triangulation

In the triangulation method [4, 23, 25], source-pivot and pivot-target bilingual corpora are used to train source-pivot and pivot-target phrase tables. Then, the source and target phrases are connected via common pivot phrases.

Given a source phrase s of the source-pivot phrase table T_{SP_s} and a target phrase t of the pivot-target phrase table T_{P_tT} , the phrase translation probability is estimated via common pivot phrases p based on the following feature function.

$$\phi(t|s) = \sum_{p \in T_{SP_s} \cap T_{P_tT}} \phi(p|s)\phi(t|p) \quad (1)$$

When the connections are based on only the surface form of pivot phrases, the model does not take advantage of full potential connections for pivot translation.

3.2 Integrating Grammatical and Morphological Information to Pivot Translation

In order to integrate grammatical and morphological information to pivot translation, we trained factored models [12] on the source-pivot and pivot target bilingual corpora. First, the part-of-speech and lemma forms of words in the pivot sides of the bilingual corpora were generate to enrich information for pivoting via pivot phrases. Since English was used for the pivot language, analyzing part-of-speech (POS tagging) and lemma forms were conducted using the Stanford CoreNLP [14]. Then, the bilingual corpora in which words of the pivot sides in the bilingual corpora contain part-of-speech tags and lemma forms were used to train source-pivot and pivot-target phrase tables. After that, source-target phrase pairs were extracted based on common pivot phrases based on Eq. 1.

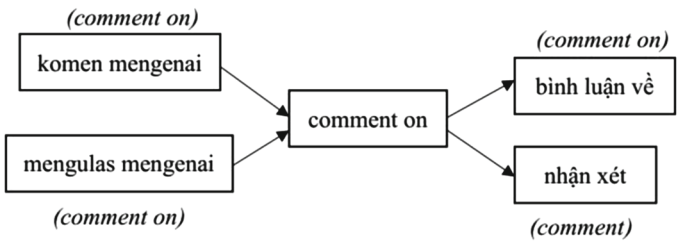
Part-Of-Speech Tags in Pivot Translation. The connections via common pivot phrases can be enriched by integrating grammatical information. As shown in Table 1, when adding grammatical information (part-of-speech tags), the pivot phrase *commented on* was divided into two cases: *commented|VBD on|IN* and

Table 1. Examples of integrating grammatical information to pivot translation. (POS tags: VBD (Verb, past tense), VBN (Verb, past participle), NN (Noun, singular or mass), VB (Verb, base form), IN (Preposition)); *Italic words*: English meaning.

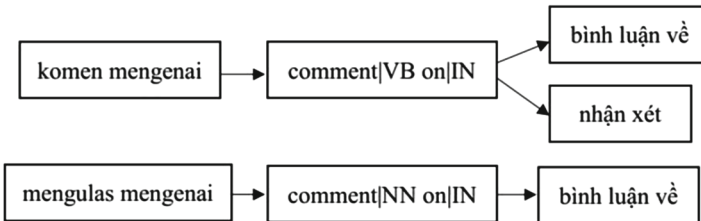
Model	Source (Malay)	Pivot (English)	Target Vietnamese
surface	mengulas mengenai <i>(comment on)</i>	commented on	bình luận về <i>(comment on)</i>
POS	mengulas mengenai <i>(comment on)</i>	commented VBD on IN	bình luận về <i>(comment on)</i>
	mengulas mengenai <i>(comment on)</i>	commented VBN on IN	đã bình luận về <i>(commented on)</i>

commented|VBN on|IN. Due to adding the part-of-speech information, a new connection to the target phrase “ *đã bình luận về* ” (*English meaning: commented on*) was employed instead of only one connection to the target phrase as in the surface model.

Integrating grammatical information also helps to filter connections via pivot phrases. Figure 1 describes an example of using part-of-speech information (POS factor) in pivot translation that help to filter connections. For the case of pivoting via surface form of pivot phrases (Fig. 1a), the two Malay phrases *komen mengenai* and *mengulas mengenai* were both connected to the pivot phrase *comment on*. Nevertheless, when integrating POS factor (Fig. 1b), the two source



a) Pivot translation via the surface form



b) Integrating POS factor to pivot translation

Fig. 1. Example of pivoting using POS factor

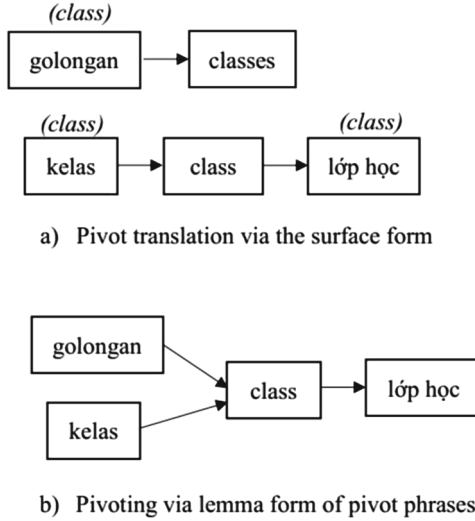


Fig. 2. Example of pivoting using lemma factor; *the italic words* indicate the English meaning

phrases were connected to two different pivot phrases (*comment|VB on|IN* and *comment|NN on|IN*). The separated connections help to classify the connections in more detailed, that refine the translation probabilities.

Lemma Forms in Pivot Translation. As shown in [12], using lemma forms can improve the problem of sparse training data in SMT. We propose connecting phrases via the lemma form instead of the surface form of pivot phrases. Figure 2 describes an example of using lemma forms in pivot translation. Because of the sparse data problem, the Malay word *golongan* cannot be connected to any Vietnamese target word. However, when using the lemma form (Fig. 2b), because the pivot words *class* and *classes* share the same lemma form (*class*), a new informative connection was generated for the source word *golongan*.

3.3 Combining Features to Pivot Translation

Because part-of-speech information and lemma forms of pivot phrases were added to improve the baseline pivot method (trained on the surface form of pivot phrases), we introduce a model that combines three following components:

- *base*: the triangulated phrase table based on the surface form of pivot phrases
- *pos*: the triangulated phrase table based on factored training using part-of-speech tags of pivot phrases
- *lem*: the triangulated phrase table based on factored training using lemma forms of pivot phrases

A combined phrase table was generated using linear interpolation [20], in which the probability of a target phrase t given a source phrase s can be computed by Eq. 2.

$$p(t|s) = \lambda_{base}p_{base}(t|s) + \lambda_{pos}p_{pos}(t|s) + \lambda_{lem}p_{lem}(t|s) \quad (2)$$

where $p_{base}(t|s)$, $p_{pos}(t|s)$, and $p_{lem}(t|s)$ stand for the translation probability of the baseline, POS tags, and lemma models, respectively.

The interpolation parameters λ_{base} , λ_{pos} , and λ_{lem} in which $\lambda_{base} + \lambda_{pos} + \lambda_{lem} = 1$ were computed by the following strategies.

- *tune*: the parameters were tuned using a development data set.
- *weights*: the parameters were set based on the ratio of the BLEU scores when using each model separately for decoding the tuning set.

4 Experiments

We conducted experiments of translation from Indonesian, Malay, and Filipino to Vietnamese, which are Southeast Asian low-resource language pairs. We used English for the pivot language, which is one of the most common language in the world.

4.1 Setup

Experiments were conducted using the well-known SMT toolkit, Moses [13]. The word alignment was trained using GIZA++ [16] with the configuration *grow-diag-final-and*. A 5-gram language model of the target language was trained using KenLM [7]. For tuning, we used the batch MIRA [2]. BLEU scores [17] were used as the metric for evaluation.

For pivot translation, we implemented the triangulation approach of Wu and Wang, 2007 [25] using Java. As reported in [6], one of the issue of the triangulation approach is the very big size of the induced pivot phrase table. Therefore, we filtered the induced phrase table by a *n-best* strategy, in which n best candidates of target phrases were selected for each source phrase. We chose $n = 10$ in our experiments to ensure the reliability of phrase pairs. We also compared our system with the TmTriangulate [9], a python implementation of the triangulation method [25].

4.2 Corpora

For training data, we used two resources: TED data [1] and the ALT corpus (Asian Language Treebank Parallel Corpus) [22]. A multilingual parallel data set was collected from the TED talks² for training data of Indonesian, Malay, Filipino, English, and Vietnamese. The ALT corpus includes 20 K multilingual

² <https://wit3.fbk.eu/>.

sentences of English, Japanese, Indonesian, Filipino, Malay, Vietnamese, and some other Southeast Asian languages. We divided the ALT corpus into three sets: training (16K), development (2K) and test (2K). The training part (16K) was combined with the TED data set for training translation models. For monolingual data of Vietnamese to train a language model, we collected articles from the website <http://www.baomoi.com/>, which contains 16M sentences. Table 2 presents training sets.

Table 2. Bilingual corpora for training sets

Data	Sentences
Malay-English	31,608
Indonesian-English	244,858
Filipino-English	21,951
English-Vietnamese	377,736

4.3 Results

Experimental results are presented in Table 3. There are several findings from the experimental results. First, using additional knowledge of part-of-speech and lemma forms improved the baseline triangulation trained on the surface form of pivot phrases. Second, combining the baseline model (surface form) with the pos (part-of-speech) and lemma models by using the interpolation method improved the baseline model. Specifically, combining the baseline model with the pos and lemma models improved from 0.1 to 0.5 BLEU scores for the Malay-Vietnamese experiments (Table 3). For Indonesian-Vietnamese, the highest improvement comes from the combination of the baseline and lemma models (+0.35 BLEU score). Unexpectedly, for the Filipino-Vietnamese experiments, the combination of all models: baseline, pos, and lemma does not show any improvement. Meanwhile, the baseline-pos combination model slightly improved the baseline model (+0.14 BLEU).

For the comparison of our system with the TmTriangulate, our system with the filtering strategy showed much better performance.

4.4 Analysis

In this subsection, we analyze the effect of adding grammatical and morphological information in improving pivot translation. Additionally, we conducted statistical significance test to verify the improvement of the proposed methods.

Factored Models. Because of adding knowledge of part-of-speech and lemma, the number of phrase pairs in the combined phrase tables was significantly

Table 3. Experimental results (BLEU); **baseline-pos-lemma**: the interpolation model of the baseline (pivot via surface form), pos (pivot via factored models of part-of-speech), and lemma (pivot via factored models of lemma). **weights**, **tune**: the interpolation settings presented in Sect. 3.3.

Model	Dev.	Test
Malay-Vietnamese		
TmTriangulate	25.06	35.09
baseline	25.78	35.86
pos	25.82	35.99 (+0.13)
lemma	24.93	35.62
baseline-pos (weights)	25.89	36.04 (+0.18)
baseline-pos (tune)	25.89	36.08 (+0.22)
baseline-lemma (weights)	25.76	36.20 (+0.34)
baseline-lemma (tune)	25.79	36.12 (+0.26)
pos-lemma (weights)	25.72	36.11 (+0.25)
pos-lemma (tune)	25.83	36.19 (+0.33)
baseline-pos-lemma (weights)	25.81	36.38 (+0.52)
baseline-pos-lemma (tune)	25.89	36.25 (+0.39)
Indonesian-Vietnamese		
TmTriangulate	24.60	32.83
baseline	25.51	33.83
pos	25.54	33.87 (+0.04)
lemma	24.68	32.89 (+0.06)
baseline-pos (weights)	25.65	33.91 (+0.08)
baseline-pos (tune)	25.62	33.87 (+0.04)
baseline-lemma (weights)	25.38	34.07 (+0.24)
baseline-lemma (tune)	25.48	34.18 (+0.35)
pos-lemma (weights)	25.38	33.87 (+0.04)
pos-lemma (tune)	25.53	34.01 (+0.18)
baseline-pos-lemma (weights)	25.51	34.05 (+0.22)
baseline-pos-lemma (tune)	25.64	33.94 (+0.11)
Filipino-Vietnamese		
TmTriangulate	17.51	25.29
baseline	18.07	26.02
pos	18.22	25.95
lemma	17.33	25.38
baseline-pos (weights)	18.16	26.16 (+0.14)
baseline-pos (tune)	18.17	25.98
baseline-lemma (weights)	17.96	25.90
baseline-lemma (tune)	18.09	25.89
pos-lemma (weights)	18.00	25.93
pos-lemma (tune)	18.12	25.96
baseline-pos-lemma (weights)	18.16	26.00
baseline-pos-lemma (tune)	18.19	26.01

Table 4. Input factored phrase tables (**Src-Pvt**, **Pvt-Trg**: source-pivot, pivot-target phrase pairs, **Common**: common pivot phrases)

Model	Src-Pvt	Pvt-Trg	Common
Malay-Vietnamese			
baseline	83,914	395,983	33,573
pos	89,424	420,790	36,404
lemma	86,359	405,051	29,585
Indonesian-Vietnamese			
baseline	239,172	395,983	63,596
pos	250,286	420,790	68,404
lemma	240,426	405,051	54,808
Filipino-Vietnamese			
baseline	79,671	395,637	26,026
pos	82,151	420,441	27,517
lemma	80,265	404,712	23,172

increased, which cover larger ratio of vocabulary. This leads to an improvement on the baseline model. As presented in Table 5, the numbers of phrase pairs in the baseline models were significantly increased when adding part-of-speech and lemma information: Malay-Vietnamese (+37,207 phrase pairs, or 39.26%), Indonesian-Vietnamese (+40,699 phrase pairs, or 31.75%), and Filipino-Vietnamese (+47,938 phrase pairs, or 36.04%).

We evaluated the effect of factored models to pivot translation. The factored models enrich linguistic knowledge of part-of-speech and lemma. When using factored models for training source-pivot and pivot-target phrase tables, the numbers of input phrase pairs were increased, which lead to increase the number of common pivot phrases and the extracted phrase pairs in the induced phrase table output. Table 4 illustrates the statistics of input phrase tables using factored models.

Out of Vocabulary. We analyzed the out-of-vocabulary (OOV) ratio when using models for decoding on test sets. Using part-of-speech and lemma information enriched linguistic knowledge for the baseline model trained on only the surface form of pivot phrases. As described in Table 6, the OOV ratios of the baseline models were reduced such as Malay-Vietnamese (-1.65%), Indonesian-Vietnamese (-0.77%), and Filipino-Vietnamese (-0.79%).

Statistical Significance Tests. Since the improvement of the proposed method is still limited in some cases, we conducted statistical significance tests to verify the improvement. The significance tests were based on a method called

Table 5. Extracted phrase pairs by triangulation (**Pairs**, **Src**, **Trg**: source-target phrase pairs, source phrases, target phrases, respectively)

Model	Pairs	Src	Trg
Malay-Vietnamese			
baseline	94,776	16,936	24,868
pos	93,972	16,939	24,858
lemma	101,904	17,404	26,072
baseline-pos	112,529	17,587	26,475
baseline-lemma	125,903	17,942	27,559
pos-lemma	116,529	17,406	26,553
baseline-pos-lemma	131,983	17,942	27,742
Indonesian-Vietnamese			
baseline	128,206	20,492	27,511
pos	126,756	20,449	27,474
lemma	134,594	20,931	28,562
baseline-pos	144,375	20,760	28,865
baseline-lemma	161,985	21,148	29,917
pos-lemma	155,648	20,933	29,103
baseline-pos-lemma	168,905	21,148	30,119
Filipino-Vietnamese			
baseline	133,003	22,115	23,258
pos	131,015	22,046	23,216
lemma	143,429	22,635	24,317
baseline-pos	150,599	22,419	24,557
baseline-lemma	173,227	22,901	25,661
pos-lemma	165,930	22,646	24,896
baseline-pos-lemma	180,941	22,907	25,862

bootstrap resampling in machine translation [11]. There are two techniques in this method: *broad sample* and *paired bootstrap resampling*.

Broad sample. First of all, a set of n test sets of m sentences was extracted from an original test set using a technique called *broad sample*. For instance, given an original test set of 30,000 sentences, this test set can be divided into 10 samples, in which each sample contains 3000 sentences. The sample i^{th} includes the sentences $i, i + 10, \dots, i + 29990$. This is to avoid the problem when we collect a set of 3000 sentences in sequence, certain factors may show stronger effects on some samples, but not others.

Bootstrap resampling. For each sample extracted from the *broad sample* step, a large number of test sets of size n sentences was randomly drawn with replacement. For each of the drawn test sets, two translation systems that we

Table 6. Out-Of-Vocabulary ratio (**OOV phrases**: the number of phrases in the test set that were not translated)

Model	OOV phrases	OOV ratio (%)
Malay-Vietnamese		
baseline	1,510	20.42
pos	1,481	20.05
lemma	1,432	19.35
baseline-pos-lemma (weights)	1,390	18.77
baseline-pos-lemma (tune)	1,387	18.77
Indonesian-Vietnamese		
baseline	1,180	15.40
pos	1,186	15.49
lemma	1,136	14.82
baseline-pos-lemma (weights)	1,124	14.63
baseline-pos-lemma (tune)	1,159	15.12
Filipino-Vietnamese		
baseline	2,628	30.24
pos	2,643	30.23
lemma	2,609	29.74
baseline-pos-lemma (weights)	2,567	29.45
baseline-pos-lemma (tune)	2,605	29.46

want to compare were used to translate the test set and compare the performance. If one system outperforms the other system 95% of the time, we draw the conclusion that the system is better with 95% statistical significance.

Results. For generating the broad samples and random test sets, we used the same test sets with the size of 2000 sentences as described in Sect. 4.2 for the original test set. Two sample sizes of 200 and 400 sentences were used to extract 10 and

Table 7. Results of statistical significance tests

System comparison	BLEU	Size 200	Size 400
ms-vi: baseline-pos-lemma (weights) better than baseline	0.52%	40%	60%
id-vi: baseline-lemma (tune) better than baseline	0.35%	40%	80%
fil-vi: baseline-pos (weights) better than baseline	0.14%	20%	0%

Table 8. Examples of improving pivot translation by using POS and lemma factors; **baseline**, **lemma**, **pos**: the translation generated by the baseline, lemma, pos models, respectively; the *italic phrases* indicate the phrases that were not translated by the baseline model; the **bold phrases** indicate the translation by the pos and lemma models.

Setup	Example
	Malay-Vietnamese
input	FDA berkata ia sedang mengkaji semula keputusan itu , yang mempunyai tiga puluh hari untuk mematuhinya .
baseline	FDA cho biết họ đang xem xét lại quyết định này , những người đã ba mươi ngày để <i>mematuhinya</i> .
lemma	FDA cho biết nó đang xem xét lại các quyết định , với ba mươi ngày để tuân theo .
reference	FDA cho biết đang xem xét lại phán quyết này , và có ba mươi ngày để tuân theo .
meaning	the FDA says it is reviewing the ruling , which it has thirty days to comply with .
input	beliau dijangka hadir di mahkamah juvena minggu depan .
baseline	ông ta dự kiến sẽ có mặt tại tòa án <i>juvana</i> vào tuần tới .
pos	ông ta dự kiến sẽ có mặt tại tòa án thiếu niên vào tuần tới .
reference	cậu bé dự kiến sẽ xuất hiện tại tòa án vị thành niên vào tuần tới .
meaning	he is expected to appear in juvenile court next week .
	Indonesian-Vietnamese
input	menurut estimasi serikat , antara tahun 2006 dan 2007 hampir 250 penambang tewas dalam kecelakaan .
baseline	theo <i>estimasi</i> công đoàn , giữa năm 2006 và năm 2007 gần 250 thợ mỏ đã thiệt mạng trong vụ tai nạn .
lemma	theo ước tính của công đoàn , giữa năm 2006 và năm 2007 gần 250 thợ mỏ đã thiệt mạng trong vụ tai nạn .
reference	theo ước tính của công đoàn , giữa năm 2006 và 2007 gần 250 thợ mỏ đã thiệt mạng trong các vụ tai nạn .
meaning	according to union estimations , between 2006 and 2007 nearly 250 miners died in accidents .
	Filipino-Vietnamese
input	ang linya sa pagitan ng York at Leeds ay isinara ng ilang oras , na nagpaantala sa ibang serbisyo .
baseline	tuyến đường giữa của York và Leeds đã bị đóng cửa trong nhiều giờ , <i>nagpaantala</i> khác của dịch vụ .
pos	theo đường giữa của York và Leeds đã bị đóng cửa trong nhiều giờ , trì hoãn của các dịch vụ khác .
reference	tuyến đường giữa York và Leeds bị chặn trong nhiều giờ , làm đình trệ nhiều dịch vụ khác .
meaning	the line between York and Leeds was closed for several hours , delaying other services .

5 samples respectively using the broad sample technique. Then, for each sample, a large number of 100 test sets were randomly drawn with replacement. Experimental results are presented in Table 7. For Indonesian-Vietnamese, although the improvement on the original test set was still limited (0.35%), for 80% of samples (size 400) and 40% of samples (size 200) we draw the conclusion the proposed system is better than the baseline system with at least 95% statistical significance. For Filipino-Vietnamese, for 20% of samples (2/10 samples size 200) we draw the conclusion the proposed system is better than the baseline system with 95% statistical significance. For all cases, no wrong conclusion was drawn, in which the proposed system does not improve the baseline system.

Sample Translations. We describe examples of using part-of-speech and lemma information in improving pivot translation in Table 8. Some phrases that were not translated by the baseline model can be translated by the proposed models using part-of-speech and lemma information.

5 Conclusion

In this work, we propose a method to improve pivot translation using grammatical and morphology information. Part-of-speech tags and lemma forms were added for the triangulation method instead of using only the surface form of pivot phrases. Experiments were conducted on several Southeast Asian low-resource language pairs: Indonesian-Vietnamese, Malay-Vietnamese, Filipino-Vietnamese. Experimental results showed that integrating part-of-speech and lemma information improved the triangulation method trained on the surface form of pivot phrases by 0.52 BLEU points. This showed the effectiveness of integrating grammatical and morphological information in pivot translation.

References

1. Cettolo, M., Girardi, C., Federico, M.: WIT3: web inventory of transcribed and translated talks. In: Proceedings of EAMT, pp. 261–268 (2012)
2. Cherry, C., Foster, G.: Batch tuning strategies for statistical machine translation. In: Proceedings of HLT/NAACL, pp. 427–436. Association for Computational Linguistics (2012)
3. Chu, C., Nakazawa, T., Kurohashi, S.: Constructing a Chinese-Japanese parallel corpus from Wikipedia. In: Proceedings of LREC, pp. 642–647 (2014)
4. Cohn, T., Lapata, M.: Machine translation by triangulation: making effective use of multi-parallel corpora. In: Proceedings of ACL, pp. 728–735. Association for Computational Linguistics, June 2007
5. De Gispert, A., Marino, J.B.: Catalan-English statistical machine translation without parallel corpus: bridging through Spanish. In: Proceedings of LREC, pp. 65–68. Citeseer (2006)
6. El Kholly, A., Habash, N., Leusch, G., Matusov, E., Sawaf, H.: Language independent connectivity strength features for phrase pivot statistical machine translation. In: Proceedings of ACL, pp. 412–418. Association for Computational Linguistics (2013)


7. Heafield, K.: KenLM: Faster and smaller language model queries. In: Proceedings of the Sixth Workshop on Statistical Machine Translation, pp. 187–197. Association for Computational Linguistics (2011)
8. Hewavitharana, S., Vogel, S.: Extracting parallel phrases from comparable data. In: Sharoff, S., Rapp, R., Zweigenbaum, P., Fung, P. (eds.) Building and Using Comparable Corpora, pp. 191–204. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-20128-8_10
9. Hoang, D.T., Bojar, O.: Tmtriangulate: a tool for phrase table triangulation. Prague Bull. Math. Linguist. **104**(1), 75–86 (2015)
10. Irvine, A.: Statistical machine translation in low resource settings. In: Proceedings of HLT/NAACL, pp. 54–61. Association for Computational Linguistics (2013)
11. Koehn, P.: Statistical significance tests for machine translation evaluation. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 388–395 (2004)
12. Koehn, P., Hoang, H.: Factored translation models. In: EMNLP-CoNLL, pp. 868–876 (2007)
13. Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., et al.: Moses: open source toolkit for statistical machine translation. In: Proceedings of ACL, pp. 177–180. Association for Computational Linguistics (2007)
14. Manning, C.D., Surdeanu, M., Bauer, J., Finkel, J.R., Bethard, S., McClosky, D.: The stanford coreNLP natural language processing toolkit. In: ACL (System Demonstrations), pp. 55–60 (2014)
15. Nuhn, M., Mauser, A., Ney, H.: Deciphering foreign language by combining language models and context vectors. In: Proceedings of ACL, pp. 156–164. Association for Computational Linguistics (2012)
16. Och, F.J., Ney, H.: A systematic comparison of various statistical alignment models. Comput. Linguist. **29**(1), 19–51 (2003)
17. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: Bleu: a method for automatic evaluation of machine translation. In: Proceedings of ACL, pp. 311–318. Association for Computational Linguistics (2002)
18. Ravi, S., Knight, K.: Deciphering foreign language. In: Proceedings of ACL: Human Language Technologies-Volume 1, pp. 12–21. Association for Computational Linguistics (2011)
19. Saluja, A., Hassan, H., Toutanova, K., Quirk, C.: Graph-based semi-supervised learning of translation models from monolingual data. In: Proceedings of ACL, pp. 676–686. Association for Computational Linguistics (2014)
20. Sennrich, R.: Perplexity minimization for translation model domain adaptation in statistical machine translation. In: Proceedings of EAMT, pp. 539–549 (2012)
21. Smith, J.R., Quirk, C., Toutanova, K.: Extracting parallel sentences from comparable corpora using document level alignment. In: Proceedings of HLT/NAACL, pp. 403–411. Association for Computational Linguistics (2010)
22. Thu, Y.K., Pa, W.P., Utiyama, M., Finch, A., Sumita, E.: Introducing the Asian Language Treebank (ALT). In: Proceedings of LREC, pp. 1574–1578 (2016)
23. Utiyama, M., Isahara, H.: A comparison of pivot methods for phrase-based statistical machine translation. In: Proceedings of HLT/NAACL, pp. 484–491. Association for Computational Linguistics (April 2007)
24. Wang, P., Nakov, P., Ng, H.T.: Source language adaptation approaches for resource-poor machine translation. Comput. Linguist. **42**, 277–306 (2016)

25. Wu, H., Wang, H.: Pivot language approach for phrase-based statistical machine translation. In: Proceedings of ACL, pp. 856–863. Association for Computational Linguistics, June 2007
26. Zhu, X., He, Z., Wu, H., Wang, H., Zhu, C., Zhao, T.: Improving pivot-based statistical machine translation using random walk. In: Proceedings of EMNLP, pp. 524–534. Association for Computational Linguistics, October 2013
27. Zhu, X., He, Z., Wu, H., Zhu, C., Wang, H., Zhao, T.: Improving pivot-based statistical machine translation by pivoting the co-occurrence count of phrase pairs. In: Proceedings of EMNLP, pp. 1665–1675. Association for Computational Linguistics (2014)

Corpora and Corpus-Based Language Processing



Information-Structure Annotation of the “Balanced Corpus of Contemporary Written Japanese”

Takuya Miyuchi^{1,2} , Masayuki Asahara³, Natsuko Nakagawa^{2,4}, and Sachi Kato³

¹ Graduate School of Global Studies, Tokyo University of Foreign Studies, Tokyo, Japan
miyuuchi.takuya.k0@tufs.ac.jp

² Japan Society for the Promotion of Science, Tokyo, Japan
nakagawanatuko@gmail.com

³ Center for Corpus Development,
National Institute for Japanese Language and Linguistics, Tokyo, Japan
{masayu-a,yasuda-s}@ninjal.ac.jp

⁴ Graduate School of Advanced Integration Science, Chiba University, Chiba, Japan

Abstract. The Japanese language is written without the use of articles. Therefore, when translating from Japanese to languages with articles by either humans or machines, issues in article selection arise, which are affected by the information structure (definiteness, specificity, and others) of the source language. This paper presents the annotation data of the information structure (information status, commonness, definiteness, specificity, animacy, sentience, agentivity) of the “Balanced Corpus of Contemporary Written Japanese,” to address article selection issues in translation. We present the annotation schema and statistics. Evaluation using the Kappa value demonstrates that there is a correspondence between the information status, commonness, definiteness, and specificity. Thus, we conclude that these grammatical labels affect article selection.

Keywords: Information structure · Annotation · Noun phrase
Article selection · Japanese

1 Introduction

It is difficult for people with mother tongues, which do not use articles, to learn other languages with articles [9, 19]. Information structures (IS), such as the definiteness and specificity, affect article selection. In English, article selection is based on definiteness, whereas in the Samoan language, it is based on specificity [14]. Article selection in the Kove language is based on both these characteristics [17].

Researchers in the NLP field have proposed a statistical model to detect the article errors made by English learners. The model was constructed based on substantial text produced by English native speakers [15]. However, when we consider the choice of articles in some of the languages generated by native Japanese speakers, careful attention must be paid to the IS in the Japanese noun phrases (NPs). Moreover, in machine

translation, when Japanese text is translated into languages that use articles, the IS of the Japanese NPs is critical.

This paper reports the results of annotating the grammatical information, considering the IS of the NPs in texts of “The Balanced Corpus of Contemporary Written Japanese” (BCCWJ) [13], as a basic research on article selection in machine translation.

Previously, the IS in Japanese was annotated as co-reference information. This means that only the annotation with regard to whether the information was provided in the discourse (information status) was performed. In this research, we annotate the tags for versatility, not only in terms of the information status, definiteness, and specificity but also for various points with respect to the IS.

2 Relevant Research

There are two types of IS annotations based on how the IS of the linguistic form in question is determined.

Some of the studies decide the IS depending upon the linguistic forms in question. For example, Calhoun et al. [2], referring to Vallduví and Vilkuna [20], Steedman [18] employed the prosody; the form in question with the prosody, L+H*LH%¹ is identified as a theme,² whereas that with the prosody, H*L or H*LL%³ is identified as a rheme.⁴ In addition, they annotate the information status based on whether the NP in question has been mentioned previously and whether it can be inferred from the previously mentioned entities. Hajičová et al. [5] proposed an IS annotation using the word order. This study is inspired by the Prague School tradition of the IS; hence, it identifies the linguistic form to the left of the verb as a topic. These annotation criteria are language-dependent and are not applicable to Japanese.⁵

The second type of studies employ linguistic tests. Götze et al. [4] devised a criterion for annotating the information status (given/accessible/new), the topic (aboutness topic/frame setting topic), and the focus (new-information focus/contrastive focus), independent of the language and linguistic theories. For example, the aboutness topic is determined by the following procedure [4, p. 165]:

- (1) An NP X is the aboutness topic of a sentence S containing X if
 - a. S would be a natural continuation to the announcement *Let me tell you something about X*
 - b. S would be a good answer to the question *What about X ?*
 - c. S could be naturally transformed into the sentence *Concerning X , S'* , where S' differs from S only insofar as X has been replaced by a suitable pronoun.

¹ L+H*LH% is a prosody in which the pitch rises slowly and then, the rising pitch of a speech boundary appears [18, 645–655].

² The theme approximately corresponds to a topic.

³ H*L is a prosody in which a rapid high pitch starts and falls. H*LL% is a prosody in which a low pitch follows H*L towards the end of the speech [18, 645–655].

⁴ The rheme approximately corresponds to a focus.

⁵ Calhoun et al. [2] address (spoken) English and Hajičová et al. [5] Czech, respectively.

Our study was conducted along the lines followed by Götze et al. [4]; however, it is significantly different from their study in certain respects. First, our study does not annotate the topic and focus directly. This is because the topic and focus are multi-dimensional, respectively [16]; i.e., the topic and focus can be decomposed into subdivided grammatical factors. Götze et al. [4, p. 163] distinguish various kinds of aboutness topics such as referential NPs, indefinite NPs with specific and generic interpretations, etc. It is simpler to assume that factors such as the definiteness and specificity are independent and to annotate as such. Next, more factors are known to correlate with a topic and focus: e.g., animacy and agentivity [3, 10]. Therefore, we decided to annotate these factors as part of the IS annotation.

3 Labels and Criteria

3.1 Targets and Labels

The two types of lexical items in the BCCWJ are the “short unit words” and the “long unit words.” In this research, we use the short unit words and set the labels of the following seven points shown in (2).

- (2) a. information status
 b. commonness
 c. definiteness
 d. specificity
 e. animacy
 f. sentience
 g. agentivity

3.2 Information Status

Information status depicted in (2a) is the so-called distinction between the new information and the old/given information. In some discourses, the information that the speakers want to convey to the hearers is known as the “discourse-new” and the information that the hearers already know is the “discourse-old.”⁶

- (3) a. tannin datta Ikeda Hiroko sensei wa tigatta
 homeroom_teacher COP.PST Ikeda Hiroko teacher TOP be_different-PST
 ‘The homeroom teacher, Hiroko Ikeda is different.’
 (Yomiuri [BCCWJ: PN1c_00001])
- b. sukuurukaunseraa demo atta sensei no zyugyoo wa
 school_counselor too COP.PST teacher GEN lesson TOP
 ‘The lessons of the teacher, who is also a school counselor ...’
 (Yomiuri [BCCWJ: PN1c_00001])

⁶ For details on the information status itself, see Kruijff-Korbayová and Steedman [11], Hinterwimmer [8], etc.

The underlined NP in (3a) *Ikeda Hiroko* is a discourse-new item since it appears first in this text. That in (3b) *sensei* ‘teacher’ is a discourse-old item since it refers to (3a) *Ikeda Hiroko*. These NPs are co-referred.

3.3 Commonness

Commonness, in (2b), is a parameter that expresses whether the speakers assume that the hearers already know the information. The information which the speakers think the hearers already know is “hearer-old;” whereas the information which the speakers do not think the hearers know is “hearer-new;” i.e., the hearer-old information exists but the hearer-new information does not, on the common ground of both the speakers and hearers. In addition to these two labels, we introduce a label called “bridging.” This label is given, when the NP is a bridging anaphora. Note that, when judging the commonness, the annotators may use their world knowledge.

- (4) kyantiikaidoo o nuke, oriibubatake ni kakomareta den’entitai
 Chianti_road ACC pass olive_grove by surround-PASS.PAST rural_land
 no resutoran de
 GEN restaurant in
 ‘(He) went through the Chianti road and in the restaurant surrounded by olive
 grove ...’

(Yomiuri [BCCWJ: PN4c_00001])

The first underlined NP in (4) *kyantiikaidoo* ‘Chianti road’ is a hearer-old item since it is a well-known road famous for wine, which is officially recognized as a World Heritage site, and the annotators know this road. The second NP, *oriibubatake* ‘olive grove,’ is a hearer-new item since it cannot be determined which olive grove it is, from this text. The third NP, *resutoran* ‘restaurant,’ is a bridging item since it is a kind of bridging anaphora.

3.4 Definiteness and Specificity

Definiteness, in (2c), is a semantic category that determines whether it is possible for the hearers to identify the referents.⁷ An NP, whose referents are considered by the speakers to be identifiable by the hearers is “definite;” whereas an NP, whose referents the speakers consider unidentifiable is “indefinite.” In this research, the scope of the definiteness is set to be a range of three sentences, before and after the sentence including the NP in question.

- (5) sonna usui kaban zya asobidoogu mo hairanai yo
 such thin bag TOP toy too not_contain MOD
 ‘This thin bag does not contain toys either.’

(Yomiuri [BCCWJ: PN1c_00001])

⁷ For more information regarding the definiteness, refer to Lyons [12], Heim [6].

The former underlined NP in (5) *kaban* ‘bag’ appeared three sentences previously in this text and concretely refers to the hearer’s bag itself. It naturally follows that the speaker thinks that it is identifiable by the hearer and thus it is definite. The latter, *asobidougu* ‘toy,’ is indefinite since it does not refer to any specific toy.

Specificity, in (2d), is a semantic category that determines whether the speakers consider specific referents.⁸ An NP is “specific,” when its referent is regarded to be either unique or specified by the speakers and it is “unspecific,” when its referent is not. Similar to the definiteness, the specificity scope is also set to be a range of three sentences, before and after the sentence including the NP in question.

- (6) *ikiba o usinata haitaiya ga azemiti ya naya no yokoni*
 place.to.go ACC lose-PST waste_tire NOM footpath or barn GEN side
hooti saretekita
 leaving have.been.done
 ‘The waste tires, which have nowhere to go, have been left next to footpaths or barns.’

(Hokkaido [BCCWJ: PN2e.00001])

The former underlined NP in (6) *haitaiya* ‘waste tire’ is specific since it concretely refers to the 30,000 waste tires left at Takasu-cho, Hokkaido, and this can be read within the scope. The latter, *naya* ‘barn,’ is unspecific since no specific barn is intended.

3.5 Animacy and Sentience

Animacy, in (2e), is a category that determines whether the referents are alive. Living creatures (human beings, animals, etc.) are “animate,” whereas nonliving objects (including plants) are “inanimate.” In our research, animacy tags are annotated, judging by the NP in question only. Sentience, in (2f), can be considered similar to animacy. This parameter expresses whether the referents have emotion. An NP is “sentient,” when its referent moves of its own free will and it is “insentient,” when its referent does not. For example, the choice of the verb *aru / iru* ‘exist’ in Japanese is made, based not on the distinction between animate and inanimate but on that between sentient and insentient. Thus, we need to set the parameter for sentience. Because there are cases in which the presence or absence of sentience cannot be determined from an NP only, sentience tags are annotated based on a predicate in the sentence including the NP.

- (7) *ookutibusu nado no burakkubasu rui ga, sukunakutomo 43*
 largemouth_bass such.as GEN black_bass kind NOM at_least 43
todoohuken no 761 no tameike ya kosyoo ni sinnyuusi,
 prefectures GEN 761 GEN pond or lake into intrude
 ‘Black basses such as largemouth basses intrude into at least 761 ponds or lakes in 43 prefectures.’

(Yomiuri [BCCWJ: PN4c.00001])

⁸ For more details on the specificity, see Heusinger [7].

The former underlined NP in (7) *burakkubasu* ‘black bass’ is animate and sentient. Though it is difficult to determine whether the black bass has its own free will, the predicate of it is *snnnyuusi* ‘intrude.’ This is an intentional verb that expresses intention or will, and thus, the former NP in (7) is given a sentient tag. The latter, *kosyoo* ‘lake,’ is inanimate and insentient since it refers to nonliving objects and can be easily determined not to have its own free will.

3.6 Agentivity

Agentivity, in (2g), depicts the roles that are played in a situation by those who are related to it. An NP, whose referent intentionally performs an act is an “agent” and that, whose referent undergoes a change by an act is a “patient / theme.”⁹ The agentivity tags are annotated, based on both the matrix and the subordinate clauses including the NP in question. Furthermore, we introduce the “both” and “neither” tags.

- (8) a. amikasa o kabutta hitonatukkoi egao o miru dakede
braided_hat ACC put.on-PST charming smile ACC see only
‘Only to see the charming smile, which put on a braided hat ...’
(Sankei [BCCWJ: PN1d.00001])
- b. momizi no ki ni tomatte nakayoku yorisou niwa no
maple GEN tree DAT perch chummily get_close 2-CLS GEN
kizibato
turtledove
‘two turtledoves that perch on a maple tree and chummily get close to each other’
(Sankei [BCCWJ: PN1d.00001])
- c. dokutokuna hun’iki no syasin ni narimasita
unique atmosphere GEN photo DAT become-PST
‘became a photo of unique atmosphere’
(Sankei [BCCWJ: PN1d.00001])

The underlined NP in (8a) *egao* ‘smile’ is a patient in the matrix clause, whereas it can be an agent in the subordinate clause. We annotate the “both” label on this NP. The underlined NP in (8b) *kizibato* ‘turtledove’ is given an “agent” tag, since it cannot be judged on agentivity in the matrix clause but is an agent in the subordinate clause. The underlined NP in (8c) *syasin* ‘photo’ is neither an agent nor a patient. We annotate the “neither” label on the NP.

3.7 Other Labels

When proper nouns are annotated, the annotators may consider the degree of fame and refer to their world knowledge. Expletive nouns, such as those underlined in (9a) *koto* ‘thing,’ and idiomatic expressions, such as the underlined noun in (9b) *mimi* ‘ear,’ are assigned the “expletive” and “idiomatic” tags, respectively. Note that the judgment of whether the NP in question is idiomatic depends on the annotators.

⁹ Henceforth, instead of “patient / theme,” it is referred to as a “patient” only.

Table 1. Basic label statistics

Information status	Discourse-new	Discourse-old	-	-
	1345	678	-	-
Definiteness	definite	indefinite	either	-
	1122	899	2	-
Specificity	specific	unspecific	either	neither
	1157	749	116	1
Animacy	animate	inanimate	either	-
	342	1680	1	-
Sentience	sentient	insentient	either	-
	337	1678	8	-
Agentivity	agent	patient	both	neither
	192	338	2	1491
Commonness	hearer-new	hearer-old	bridging	neither
	494	1036	489	4

Table 2. Information status and definiteness

	Discourse-new	Discourse-old
Specific	497	625
Unspecific	846	53
Either	2	0

Table 3. Information status and specificity

	Discourse-new	Discourse-old
Specific	531	626
Unspecific	705	44
Either	108	8
Neither	1	0

- (9) a. samazamana hito ga iru toiu koto ga
various person NOM exist COMP thing NOM
‘that various types of people exist ...’

(Yomiuri [BCCWJ: PN1c.00001])

- b. kiku mimi o motasete kureru n desu
listen ear ACC have-CAUS give NMLZ COP
‘(She) makes me use my listening ears.’

(Yomiuri [BCCWJ: PN1c.00001])

4 Data Statistics

The target data consists of 16 samples in the PN core data (newspaper articles) of the BCCWJ. The data includes 2,023 NPs, whose co-reference information is annotated. The annotators can check the part-of-speech (POS) information, the co-reference information, and the syntactic dependency attachment.

Table 4. Definiteness and specificity

	Definite	Indefinite	Either
Specific	1120	36	1
Unspecific	0	749	0
Either	2	113	1
Neither	0	1	0

Table 6. Definiteness and commonness

	Definite	Indefinite	Either
Hearer-old	1010	26	0
Hearer-new	74	420	0
Bridging	37	450	2
Neither	1	3	0

Table 5. Information status and commonness

	Discourse-new	Discourse-old
Hearer-old	425	611
Hearer-new	460	34
Bridging	456	33

Table 7. Specificity and commonness

	Specific	Unspecific	Either	Neither
Hearer-old	1008	26	5	0
Hearer-new	91	391	11	1
Bridging	57	332	100	0
Neither	1	3	0	0

Table 1 lists the basic statistics. The information status labels depend on the co-reference information previously annotated. The distribution of the information status differs from that of the others. Thus, the differences may affect article selection for translation from Japanese.

We demonstrate the relationship between the information status, based on the co-reference and definiteness/specificity. Table 2 is a contingency table between the information status and the definiteness. Indefinite labels appear frequently with those of the discourse-new. However, definite NPs tend to appear with both the discourse-new and the old labels. This indicates that the co-reference information may contribute to article selection, to a limited extent. The Kappa value between the discourse-new \leftrightarrow indefinite and the discourse-old \leftrightarrow definite is 0.47. Table 3 is the contingency table between the information status and the specificity. The distribution is the same as that between the information status and the definiteness. The Kappa value between the discourse-new \leftrightarrow unspecific and the discourse-old \leftrightarrow specific is 0.43. Table 4 is the contingency table between the definiteness and specificity. The unspecific NPs are all indefinite. Thus, there are no definite and unspecific NPs.

Further, we demonstrate the relationship between the definiteness/specificity and the commonness, which is the information structure in terms of the hearers. Table 5 is the contingency table between the information status and the commonness. The discourse-new NPs in the information status are uniformly distributed over the hearer-old/hearer-new/bridging. The contingency between the commonness and definiteness is shown in Table 6 and that between the commonness and specificity in Table 7, respectively. When the hearer-new and bridging labels are merged into one category, the Kappa value between the hearer-old \leftrightarrow indefinite and the hearer-new/bridging \leftrightarrow definite is 0.86. Under the same condition, the Kappa value between the hearer-old \leftrightarrow unspecific and the hearer-new/bridging \leftrightarrow specific is 0.81. This indicates that the commonness,

Table 8. Correspondence of the labels (Kappa value)

		Commonness	Definiteness	Specificity	Animacy	Sentience	Agentivity
Information status	discourse-old/ (discourse -new)	0.51	0.47	0.43	0.1	0.1	0.24
Commonness	hearer-old/(hearer- new/bridging)		0.86	0.81	-0.04	-0.04	0.25
Definiteness	definite/(indefinite)			0.96	0.04	0.04	0.37
Specificity	specific/(unspecific)				0.02	0.02	0.36
Animacy	animate/(inanimate)					0.98	0.41
Sentience	sentient/(insentient)						0.41
Agentivity	agent/(patient)						

which considers the hearers’ perspectives, is more significant for estimating the definiteness/specificity necessary for article estimation than the information status in a discourse, based on the co-reference information.

Finally, we present the evaluation of the correspondence between the other labels by the Kappa score, in Table 8. During the evaluation, the “either” and “neither” labels were excluded. The results in this Table suggest that the animacy, sentience, and agentivity do not contribute in deciding the information status and the commonness.

5 Conclusion

In this paper, we have presented the annotation data of the information structure (IS) of the BCCWJ. We have introduced seven IS concepts for the NPs in Japanese. The distributions of the annotation labels show that the co-reference information is insufficient for article selection or correction, for the Japanese; the newly introduced labels may facilitate article selection in translation.

In future, we intend to carry out the following: First, we intend to perform an experiment in which the subject participants are asked to annotate the IS. The labels, in the paper, are annotated by linguists. We plan to design linguistic questions to enable the judging of IS labels by non-linguists. To perform this task, we have to design linguistic tests that can be recognized by non-linguists. This would enable us to increase not only the number of annotations for one NP but also the target samples.

Second, we hope to develop IS estimation models based on machine-learning techniques with a thesaurus. Then, our aim is to perform the evaluation of article selection by machine translation based on the Japanese texts.

Third, we intend to analyze the IS annotation data in comparison with translated texts. A part of the data reported in this paper is translated into English, Italian, Chinese, and Indonesian by humans and the texts are open to public. The way to select articles, in human translation, can be clarified by contrasting the IS data in Japanese with the presence or absence of articles in the translated texts in English and Italian, which have articles.

Finally, we aim to overlay the IS annotation on eye tracking data [1] to investigate the relationship between the IS and the lines of sight.

Acknowledgment. This work was supported by JSPS KAKENHI Grant Numbers: JP25284083, JP17H00917, JP17J07534.

References


1. Asahara, M., Ono, H., Miyamoto, E.T.: Reading-time annotations for balanced corpus of contemporary written Japanese. In: Proceedings of COLING-2016, pp. 684–694 (2016)
2. Calhoun, S., Nissim, M., Steedman, M., Brenier, J.: A framework for annotating information structure in discourse. In: Meyers, A. (ed.) Proceedings of the Workshop on Frontiers in Corpus Annotations II: Pie in the Sky, pp. 45–52. The Association for Computational Linguistics, Ann Arbor (2005)
3. Givón, T.: Topic, pronoun, and grammatical agreement. In: Li, C.N. (ed.) Subject and Topic, pp. 149–187. Academic Press, New York (1976)
4. Götze, M., Weskott, T., Endriss, C., Fiedler, I., Hinterwimmer, S., Petrova, S., Schwarz, A., Skopeteas, S., Stoel, R.: Information structure. In: Dipper, S., Götze, M., Skopeteas, S. (eds.) Information Structure in Cross-Linguistic Corpora: Annotation Guidelines for Phonology, Morphology, Syntax, Semantics and Information Structure, vol. 7, pp. 147–187. Universitätsverlag Potsdam (2007)
5. Hajičová, E., Panevová, J., Sgall, P.: A manual for tectogrammatical tagging of the Prague Dependency Treebank. Technical report, ÚFAL/CKL, (TR-2000-09) (2000)
6. Heim, I.: Definiteness and indefiniteness. In: von Heusinger, K., Maienborn, C., Portner, P. (eds.) Semantics: An International Handbook of Natural Language Meaning, vol. 2, pp. 996–1025. Mouton de Gruyter (2011)
7. von Heusinger, K.: Specificity. In: von Heusinger, K., Maienborn, C., Portner, P. (eds.) Semantics: An International Handbook of Natural Language Meaning, vol. 2, pp. 1058–1087. Mouton de Gruyter (2011)
8. Hinterwimmer, S.: Information structure and truth-conditional semantics. In: von Heusinger, K., Maienborn, C., Portner, P. (eds.) Semantics: An International Handbook of Natural Language Meaning, vol. 2, pp. 1875–1908. Mouton de Gruyter (2011)
9. Ionin, T., Ko, H., Wexler, K.: Article semantics in L2 acquisition: the role of specificity. *Lang. Acquis.* **12**(1), 3–69 (2004)
10. Keenan, E.L.: Towards a universal definition of “subject”. In: Li, C.N. (ed.) Subject and Topic, pp. 303–334. Academic Press, New York (1976)
11. Kruijff-Korbayová, I., Steedman, M.: Discourse and information structure. *J. Logic Lang. Inf.* **12**(3), 249–259 (2003)
12. Lyons, C.: Definiteness. Cambridge University Press, Cambridge (1999)
13. Maekawa, K., Yamazaki, M., Ogiso, T., Maruyama, T., Ogura, H., Kashino, W., Koiso, H., Yamaguchi, M., Tanaka, M., Den, Y.: Balanced corpus of contemporary written Japanese. *Lang. Resour. Eval.* **48**(2), 345–371 (2014)
14. Mosel, U., Hovdhaugen, E.: Samoan Reference Grammar. Scandinavian University Press, Oslo (1992)
15. Nagata, R., Iguchi, T., Masui, F., Kawai, A., Naoki, I.: A statistical model based on the three head words for detecting article errors. *IEICE Trans. Inf. Syst.* **88**(7), 1700–1706 (2005)
16. Nakagawa, N.: Information structure in spoken Japanese: Particles, word order, and intonation. Ph.D. thesis, Kyoto University (2016)
17. Sato, H.: Definiteness and specificity in Kove. In: International Workshop on Information Structure of Austronesian Languages, pp. 37–45. The Research Institute for Languages and Cultures of Asia and Africa, Tokyo University of Foreign Studies, Tokyo (2013)
18. Steedman, M.: Information structure and the syntax-phonology interface. *Linguist. Inq.* **34**, 649–689 (2000)

19. Tanaka, J.: A multivariate analysis of L2 English article use by article-less L1 learners. In: Voss, E., Tai, S.J.D., Li, Z. (eds.) *Selected Proceedings of the 2011 Second Language Research Forum*, pp. 139–147. Cascadia Press, Somerville (2013)
20. Vallduví, E., Vilkkuna, M.: On rheme and kontrast. In: Culicover, P.W., McNally, L. (eds.) *The Limits of Syntax*, pp. 79–108. Academic Press, San Diego (1998)

Syntax and Syntactic Analysis



Khmer POS Tagging Using Conditional Random Fields

Sokunsatya Sangvat  and Charnyote Pluempitiwiriyaewj ^(✉)

Faculty of Information and Communication Technology, Mahidol University,
Salaya, Nakhon Pathom, Thailand

sangvat.sok@student.mahidol.ac.th, charnyote.plu@mahidol.ac.th

Abstract. The transformation-based approach with hybrid of rule-based and trigram have already been introduced for Khmer part-of-speech (POS) tagging. In this study, in order to further explore this topic, we present an alternative approach to Khmer POS tagging using Conditional Random Fields (CRFs). Since the features greatly affect the tagging accuracy, we investigate five groups of features and use them with the CRF model. First, we study different contextual information and use it as our baseline model. We then analyze the characteristics of Khmer and come up with three additional groups of language-related features including morphemes, word-shapes and name-entities. We also explore the use of lexicon as features to further improve the accuracy of our tagger. Our proposed approach has been evaluated on a corpus of 41,058 words and 27 POS tags. The comparative study has shown that our proposed approach produces a competitive accuracy compared to other Khmer POS tagging approaches.

Keywords: Khmer · Part-of-speech tagging · POS tagging
Conditional Random Fields

1 Introduction

Part-of-speech (POS) tagging, a process of assigning each word in a text with the most suitable part-of-speech or lexical category, is an important task in natural language processing. It is used as a pre-processing step in many applications such as information retrieval/extraction, machine translation and speech recognition.

Many work have been published on POS tagging for many different languages. However, the research of POS tagging for Khmer, the official language of Cambodia, is currently in early stage. Based on our study, there is only one published work that discusses the topic [1]. The development of Khmer POS tagging is necessary as it would have a significant impact on others NLP researches and applications of Khmer. Therefore, the problem of Khmer POS tagging should be further explored.

In this paper, we present a new approach for automatic Khmer POS tagging using Condition Random Fields (CRFs). Because the features have great impacts on the tagging accuracy, we introduce four groups of features for the CRF model. First, we investigate different templates of contextual information and use it as our baseline model. Then, we investigate some characteristics of Khmer and come up with three

additional groups of language-related features including morphemes, word-shapes and name-entities. In addition, we also explore the use of lexicon to further improve the accuracy of our tagger. Our POS tagger has been developed and evaluated on a corpus of 41,058 words and 27 POS tags that have been borrowed from the previous work in Khmer POS tagging conducted by Nou and Kameyama [1]. Our comparative study has shown that our proposed approach produces a competitive accuracy compared to Nou's and Kameyama's [1] approach.

2 Related Work

A number of statistical models have been proposed for English POS tagging. A set of top performing models includes transformation-based approach, support vector machine, maximum entropy and decision tree. These models achieve from 96% to 97% accuracy [2-5].

An elastic-input neuro tagger [6] and a hybrid tagger, combined with a neural network and Brill's error-driven learning [7] have been proposed for Thai, a language which shares a considerable number of vocabularies and grammar rules with Khmer. The two models achieve 94.4% and 95.5% accuracies respectively. Another study [8] has developed a more accurate tagger using support vector machine which increased the precision up to 96.1%.

A genetic-algorithm-based POS tagger [9] has been introduced for Chinese, which has overlapping grammar structures with Khmer. The proposed model has been shown to be more flexible to incorporate both statistical and rule information than other models such as recurrent neural net, hidden markov model and rule-based model. As a result, genetic algorithm achieves 95.8% accuracy which is higher than the other models. The maximum entropy model has also been proposed for Chinese POS tagging [10]. The model has been built using a large annotated corpus and produces 96.8% accuracy.

The most related work to our work was conducted by Nou and Kameyama [1]. Nou and Kameyama [1] have proposed a transformation-based approach with hybrid of rule-based and tri-gram for Khmer POS tagging. The problem is separated into two tasks: known words and unknown words handling. The known word handling adopted the transformation-based learning approach [2]. In this approach, the tagger first assigns the most frequent POS to each word by looking up in the lexicon (dictionary) and then applies the learned rules, obtained from the training data, to reduce the error. However, the tagger often encounters a number of unknown words, which do not appear in the lexicon and the training data. Thus, the tagging accuracy can be low because the rules of those words cannot be obtained. Therefore, the unknown words are handled in the second task which uses a hybrid of rule-based approach and tri-grams. The rule-based approach tags the words based on their internal structure while the tri-grams relies on contextual information. The hybrid of both models combines the strength of each other and thus produces a higher accuracy. Overall, the proposed approach achieves 95.27% and 91.96% of recall on the training and the testing set respectively.

3 Overview of Conditional Random Fields

Conditional Random Fields (CRFs), introduced by [11], are conditional probabilistic distribution models, designed for the problem of sequence labelling. They are a type of undirected graphical model that computes a single log-linear distribution over a state sequence (e.g. POS tags sequence) given an observation sequence (e.g. words in a sentence). The conditional probability of a state sequence $S = (s_1, s_2, s_3, \dots, s_T)$ given an observation sequence $O = (o_1, o_2, o_3, \dots, o_T)$ is defined as:

$$P(S|O) = \frac{1}{Z_0} \exp\left(\sum_{t=1}^T \sum_{k=1}^K \lambda_k f_k(s_{t-1}, s_t, o, t)\right) \tag{1}$$

where $f_k(s_{t-1}, s_t, o, t)$ is a feature function whose values could have a huge range, but they are typically binary. There are totally K feature functions. Each feature function is associated with λ , a learned weight computed using the training data. Z_0 is the normalization factor which is used to make all conditional probability of all candidate paths sum up to 1. Thus, the normalization factor is calculated as:

$$Z_0 = \sum_s \exp\left(\sum_{t=1}^T \sum_{k=1}^K \lambda_k f_k(s_{t-1}, s_t, o, t)\right) \tag{2}$$

We are given training data $D = \{s^{(i)}, o^{(i)}\}_{i=1}^N$, where each $o^{(i)} = \{o_1^{(i)}, o_2^{(i)}, \dots, o_T^{(i)}\}$ is an observation sequence, and each $s^{(i)} = \{s_1^{(i)}, s_2^{(i)}, \dots, s_T^{(i)}\}$ is a state sequence. The CRF model is trained by using the standard maximum log-likelihood estimation method. In order to avoid overfitting, we use regularization which penalizes the weight vectors whose norm is too large. Thus, the log-likelihood L of a given training data D is calculated as:

$$L = \sum_{i=1}^N \log(P(s^{(i)}|o^{(i)})) - \sum_{k=1}^K \frac{\lambda_k^2}{2\sigma^2} \tag{3}$$

In the second sum, the parameter λ is set to maximize the log-likelihood L by using various training algorithms. The parameter σ^2 is a free parameter which controls how much the weights are penalized.

We use CRFsuite [12], an open-source implementation of CRFs, for our experiment. The CRF model is trained using Gradient Descent algorithm with the L-BFGS method. We try this training method with different parameter values and find the best result by setting all parameters with the default values.

4 Khmer and Features Analysis

Khmer is the official language of Cambodia. It is spoken by approximately 16 million people of Cambodia, Khmer ethnicity in southern Vietnam and northeastern Thailand. Khmer is largely influenced by Pali and Sanskrit through Hinduism and Buddhism.

In this study, the POS tagger has been built by considering a number of Khmer characteristics which are the following:

- Khmer text are written continuously without explicit word boundary. Text need to be segmented before assigning POS.
- A word in Khmer can have more than one possible part-of-speeches depending on the meaning and the context of the word. For example, the word “មុន” /m’un/ can be used as an adverb with the meaning of “before” or as a noun with the meaning of “acne”.
- Most of the words that are borrowed from Pali and Sanskrit have more than one acceptable spellings. They can be written in both Pali/Sanskrit or modern simplified way. e.g. The Khmer word for “value” can be written as “តម្លៃ” /t’aml’ai/(Pali/Sanskrit) or “តំលៃ” /t’aml’ai/ (modern simplified).
- Adding prefixes or suffixes to a word can change the POS of the word. For example, adding the prefix “អ្នក” /n’a’k/ to the word “បើកបរ” /b’oekbr/ Drive [Verb] will produce the word “អ្នកបើកបរ” /na:kb’oekbr/ Driver [Noun].
- A considerable number of words are compound words, which are the combinations of multiple root words. The POS of the compound word is dependent on the POS of the root words. For example, the compound noun “ក្រុមហ៊ុន” /kr’umh’un/ (Company) [Noun] is a combination of the word “ក្រុម” /kr’um/ (Group) [Noun] and the word “ហ៊ុន” /h’un/ (Investment) [Noun].

We analyze the characteristics of Khmer and come up with features that are necessary for Khmer POS tagging. The features are categorized into four groups: contextual, morphological, word-shape and lexical features.

4.1 Contextual Features

Contextual features of a particular word in a sentence are defined with respect to a window of its surrounding words (proceeding words and succeeding words) and their combinations. The contextual features play an important role in producing high tagging accuracy because the POS tag of a word is dependent on the word itself and the surrounding context. However, it is challenging to come up with the template of contextual features that work best for Khmer POS tagging. Therefore, we conduct an experiment on three pre-defined templates of contextual features as shown in Table 1. The first template is an example that comes with in CRFsuite tool [12]. It contains five word-features and two two-words-transition features. The second template extends the first template by adding three three-words-transition features. The third template extends the second template by adding a five-words-transition feature. The objective of the experiment is find the most suitable template of contextual features for Khmer POS tagging.

Table 1. Templates of contextual features where $w[i]$ ($w[-i]$) is the i -th word to the right (left) of the current word $w[0]$

Template	Features
1	The word features: $w[-2], w[-1], w[0], w[1], w[2]$ The words-transition features: $(w[-1]/w[0]), (w[0]/w[1])$
2	The word features: $w[-2], w[-1], w[0], w[1], w[2]$ The words-transition features: $(w[-1]/w[0]), (w[0]/w[1]),$ $(w[-2]/w[-1]/w[0]), (w[-1]/w[0]/w[1]), (w[0]/w[1]/w[2])$
3	The word features: $w[-2], w[-1], w[0], w[1], w[2]$ The words-transition features: $(w[-1]/w[0]), (w[0]/w[1]),$ $(w[-2]/w[-1]/w[0]), (w[-1]/w[0]/w[1]), (w[0]/w[1]/w[2]),$ $(w[-2]/w[-1]/w[0]/w[1]/w[2])$

4.2 Morphological Features

A significant number of Khmer words are formed using morphological processes including prefixes, suffixes and compounding. With respect to the morphological rules of Khmer, we come up with the following morphological features that are useful for predicting POS.

- Prefixes and Suffixes: The POS of a Khmer word can be changed by adding a prefix (e.g., ផ្អែក /'n'a'k/, ភ័យ /bh'āb/, ក្រែង /k'ār/, ...) or a suffix (e.g., ជន /jn/, កិច្ច /k'icc/, ធ្វើ /dhmr/, ...). However, it is challenging to define all prefixes and suffixes in Khmer. Therefore, we extract fixed numbers (from 2 to 6) of characters from each word and use them as prefix and suffix features.
- Compound Words: The POS of a compound word can be predicted from the POS of the root words. We propose five features corresponding to the five most common types of compound words: noun-noun, noun-verb, verb-verb, adjective-adjective and adverb-adverb. Each feature check whether or not a word is one of the compounding types. For example, the word “ក្រុមហ៊ុន” /kr'umh'un/ (Company) [Noun], a combination of the word “ក្រុម” /kr'um/ (Group) [Noun] and the word “ហ៊ុន” /h'un/ (Investment) [Noun], is a noun-noun-compounding type and not the others.

4.3 Word-Shape Features

Word-shape features check whether or not a word matches a particular pattern. For example, abbreviation in English is often written with all capital letters. We have studied the patterns of Khmer words and introduce four different patterns that could predict four

different POS tags: cardinal number (ឃុំ.សំ.), ordinal number (ឃុំ.សំ.), abbreviation (អ.ក.) and foreign words (ឃុំ.វ.).

- Cardinal number: It is identified by having each character as Khmer number 0 to 9 (០, ១, ២, ៣, ៤, ៥, ៦, ៧, ៨, ៩). For example, “១២” /dɔ:b pi:/ (12) is a cardinal number.
- Ordinal number: It is identified by having the word “ទី” /dʔ/ (at) followed by Khmer number 0 to 9 (០, ១, ២, ៣, ៤, ៥, ៦, ៧, ៨, ៩). For example, “ទី១២” /dʔ dɔ:b pi:/ (12th) is an ordinal number.
- Abbreviation: It is identified by having two to five consonants separated by period “.”. For example, “អ.ស.ប.” is the abbreviation for “អង្គការ សហ ប្រជាជាតិ” /a'ngk'a s'ahh'ah br'a'jr'aj'at/ (United Nations).
- Foreign words: It is identified if it has a non-Khmer character.

4.4 Name-Entity Features

In Khmer, name entities such as names of people, places and organizations are difficult to tag. Unlike English, Khmer does not have capitalized structure to identify name-entities. Moreover, there a significant number of people names that overlap with Khmer words. To help the model identify name-entities, we propose two features which are described as follows:

- Gazetteer-lookup feature: We have built 3 gazetteer lists (lists of name entities): people, places and organization lists. The three lists are extracted from KCorpus released by PAN Localization of Cambodia [13]. However, the names that also belong to other POS tags are excluded from the lists. For example, “លោក” is a very popular Khmer name, but it is also an adjective meaning “lucky”. The word “លោក” is excluded from the gazetteer lists. After the gazetteer lists are built, the feature checks whether or not a particular word appears in the gazetteer lists.
- Tittle feature: Most name-entities in Khmer are preceded by some common titles such as លោក /l'ok/ (Mr), កញ្ញា /kññ'a/ (Miss), ខេត្ត /kh'ett/ (Province), ទីក្រុង /t'kr'ug/ (City), ក្រុមហ៊ុន /kr'umh'un/ (Company), ..., etc. This feature checks whether or not a particular word is followed by one of the title.

4.5 Lexical Features

Lexical features of a particular word are defined with respect to all of its possible POS(es) obtained from a lexicon (dictionary). In this work, we use the lexicon built by [1]. The lexicon contains approximately 32,000 words extracted from Royal Academy wordlist which has been approved by the Royal government of Cambodia. The lexicon includes most of the Khmer root words and their corresponding POS(es). The lexicon is used in two ways. First, all possible POS(es) of each word are used as features to CRFs. This will help the model predicts the POS of the words which do not appear in training set. Second, the words in the lexicon are used as root words in the compounding features described in Sect. 4.2.

5 Experimental Setup and Results

5.1 Experimental Setup

To be able to compare the performance with [1], we have set up the experiment that uses the same corpus. The corpus totally contains 41,058 words (1,298 sentences) and is tagged with 27 POS classes.

We adopt five-fold cross validation to evaluate our approach. The corpus is divided into five equal sized subsets. One of the five subset is used as the testing data and the remaining four subsets are used as the training data. The process is repeated for five iterations, where each of the five subsets is used once as the testing data. The final accuracy is the average accuracy across all five iterations. The accuracy is calculated as:

$$Accuracy = \frac{\text{Number of correctly tagged words}}{\text{Total number of words}} \quad (4)$$

In this study, we perform three experiments. The objective of the first experiment is to find the most suitable template of contextual features. The objective of the second experiment is to evaluate the accuracy of our proposed features. The objective of the third experiment is to compare our proposed approach to [1] in term of accuracy.

5.2 Finding the Most Suitable Template of Contextual Features

As described in Sect. 4.1, it is challenging to come up with the template of contextual features that works best for Khmer POS tagging. Therefore, we define three templates of contextual features (as shown in Table 1) and test the accuracy of each one. With the five-fold cross validation, the tagging accuracy on testing set performed by each template is shown in Table 2. We can see that Template 1, which is an example that come with the CRFsuite [12], produces the highest accuracy. Therefore, the contextual features defined in Template 1 will be used as the baseline features for the rest of the experiments.

Table 2. Accuracy of different templates of contextual features

Features	Testing accuracy (%)
Template 1	86.54
Template 2	85.76
Template 3	85.48

5.3 Features Evaluation

To improve the accuracy upon the contextual features (the baseline features), the morphological, the word-shape, the name-entity and the lexical features are added. Table 3 shows the accuracy of continuous adding the morphological (M), the word-shape (WS), the name-entity (NE) and the lexical (L) features to the contextual features (C). As we can see, adding the morphological and the lexical features to the model significantly improves the accuracy by 5.64% and 3.12% respectively while adding the

word-shape and name-entity features slightly increases the accuracy by 0.23% and 0.86% respectively. In general, the testing accuracy improves as we keep on adding the features to the model. From this observation, we can conclude that the uses of the morphological, the word-shape, the name-entity and the lexical features are effective in improving the tagging accuracy together with the contextual features. Our proposed feature set (C + M + WS + NE + L) produces 96.39% accuracy.

Table 3. Tagging accuracy of accumulative features starting with the baseline

Features	Testing accuracy (%)	Δ Accuracy (%)
C (baseline)	86.54	–
C + M	92.18	+5.64
C + M + WS	92.41	+0.23
C + M + WS + NE	93.27	+0.86
C + M + WS + NE + L	96.39	+3.12

5.4 Comparison to the Previous Work

We evaluate the accuracy of our proposed approach and compare the results with Nou and Kameyama [1]. Our proposed approach is evaluated in two separated cases. In the first case, we adopt the same approach as [1] in order to fairly compare the accuracy. The corpus is divided into two sets: 32,088 words for training set and 8,970 words for testing set. All the training data are extracted from Kohsantepheap newspaper (a very popular newspaper in Cambodia). 60% of the testing data are extracted from the same newspaper and the other 40% are from letters, legends, and reading articles in high school student’s textbook. We call this case hold-out validation. In the second case, using the same corpus, we evaluate our approach using five-fold cross validation. In each fold, 80% of the corpus is used for the training and the rest 20% is used for testing. Both training and testing data in each fold are extracted from mixed domains.

Table 4 compares testing accuracy between our proposed approach and Nou’s and Kameyama’s [1] approach. As can be seen, the accuracy of our proposed approach is competitive compared to Nou’s and Kameyama’s [1] approach when we adopt the hold-out validation. The accuracy is even higher when adopting the five-fold cross validation. In this comparison, we only consider the experimental results in which Nou and Kameyama [1] use the complete lexicon.

Table 4. Comparison between our proposed approach and Nou’s and Kameyama’s [1] approach

Method	Testing accuracy
Nou’s and Kameyama’s approach [1]	95.10%
Our proposed approach (hold-out validation)	95.52%
Our proposed approach (5-fold cross validation)	96.39%

6 Conclusion and Future Work

In this study, we have introduced a new approach to Khmer POS tagging using Conditional Random Fields (CRFs). We have come up with 5 groups of features: contextual, morphological, word-shape, name-entity and lexical features which are used in the CRF model. These features are based on the example template that comes with CRFsuite and the study of Khmer characteristics. The experimental results have shown that they are effective in improving the tagging accuracy. Our proposed approach produces a competitive accuracy compared to earlier approaches in Khmer POS tagging.

There are several ways to increase the tagging accuracy in order to reach state-of-the-art level. One possible way is to train the tagger on a larger annotated corpus with diverse kinds of text. This would allow the model to obtain more statistical information and decrease the number of unknown words in the testing set. Another possible way is to use more advanced machine learning models such as support vector machine or deep learning. It might also be useful to combine statistical approach with rule-based approach to improve the accuracy. This work can be used to as a pre-processing step in other researches of Khmer natural language processing such as name-entity recognition, word sense disambiguation and parsing.

Acknowledgements. This research project was supported by Faculty of Information and Communication Technology, Mahidol University.

References

1. Nou, C., Kameyama, W.: Khmer POS tagger: a transformation-based approach with hybrid unknown word handling. In: International Conference on Semantic Computing (ICSC 2007), pp. 482–489 (2007). <https://doi.org/10.1109/icosc.2007.4338385>
2. Brill, E.: Transformation-based error-driven learning and natural language processing: a case study in part-of-speech tagging. *Comput. Linguist.* **21**, 543–565 (1995)
3. Giménez, J., Màrquez, L.: SVMTool: a general POS tagger generator based on support vector machines. In: Proceedings of the 4th International Conference on Language Resources and Evaluation, pp. 43–46 (2004)
4. Ratnaparkhi, A.: A maximum entropy model for part-of-speech tagging. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, vol. 1, pp. 133–142 (1996)
5. Black, E., Jelinek, F., Lafferty, J., Mercer, R., Roukos, S.: Decision tree models applied to the labeling of text with parts-of-speech. In: Proceedings of the Workshop on Speech and Natural Language, pp. 117–121 (1992). <https://doi.org/10.3115/1075527.1075554>
6. Ma, Q., Uchimoto, K., Murata, M., Isahara, H.: Elastic neural networks for part of speech tagging. In: IJCNN99 International Joint Conference on Neural Networks Proceedings, vol. 5, pp. 2991–2996 (1999). <https://doi.org/10.1109/ijcnn.1999.835997>
7. Ma, Q., Murata, M., Uchimoto, K., Isahara, H.: Hybrid neuro and rule-based part of speech taggers. In: Proceedings of the 18th Conference on Computational Linguistics, pp. 509–515 (2000). <https://doi.org/10.3115/990820.990894>

8. Murata, M., Ma, Q., Isahara, H.: Part of speech tagging in Thai language using support vector machine. In: NLPRS 2001 Workshop, The Second Workshop on Natural Language Processing and Neural Networks (NLPNN2001) (2001)
9. Lua, K.T.: Part of Speech Tagging of Chinese Sentences Using Genetic Algorithm (1996). In: Proceedings of ICC96, pp. 45–49. National University of Singapore
10. Zhao, J., Wang, X.-L.: Chinese POS tagging based on maximum entropy model. In: Proceedings International Conference on Machine Learning and Cybernetics, vol. 1, pp. 601–605 (2002). <https://doi.org/10.1109/icmlc.2002.1174406>
11. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001), pp. 282–289 (2001)
12. Okazaki, N.: CRFsuite: a fast implementation of Conditional Random Fields (CRFs) (2007). <http://www.chokkan.org/software/crfsuite/>
13. Khmer Part-of-Speech Tagger (2008). <http://www.panl10n.net/english/Outputs%20Phase%202/CCs/Cambodia/MoEYS/Papers/2008/KhmerPOSTaggingV1.0.pdf>



Statistical Khmer Name Romanization

Chenchen Ding¹(✉), Vichet Chea², Masao Utiyama¹, Eiichiro Sumita¹,
Sethserey Sam², and Sopheap Seng²

¹ Advanced Translation Technology Laboratory, ASTREC,
National Institute of Information and Communications Technology, 3-5 Hikaridai,
Seikacho, Sorakugun, Kyoto 619-0289, Japan

{chenchen.ding,mutiyamam,eiichiro.sumita}@nict.go.jp

² Research and Development Center, National Institute of Posts,
Telecommunication and ICT, #41 Russian Federation Blvd., Phnom Penh, Cambodia

{vichet.chea,sethserey.sam,sopheap.seng}@niptict.edu.kh

Abstract. We discuss and solve the task of Khmer name Romanization. Although several standard Romanization systems exist for Khmer, conventional transcription methods are applied prevalently in practice. These are inconsistent and complicated in some cases, due to unstable phonemic, orthographic, and etymological principles. Consequently, statistical approaches are required for the task. We collect and manually align 7,658 Khmer name Romanization instances. The alignment scheme is designed to reach a precise, consistent, and monotonic correspondence between the two different writing systems on grapheme level, through which various machine learning approaches are facilitated. Experimental results demonstrate that standard approaches of conditional random fields and support vector machine supervised by the manual alignment achieve a precision of .99 on grapheme level, which outperforms a state-of-the-art recurrent neural network approach in a pure sequence-to-sequence manner. The manually aligned data have been released under a license of CC BY-NC-SA for the research community.

1 Introduction

Romanization is a linguistic task used to transform a non-Latin writing system into Latin script—which is not a huge but an important and language-specific task in natural language processing (NLP), especially in statistical machine translation (SMT)—for those languages that do not use Latin script in orthography. In academia, orthographic transliteration, phonemic transcription, or mixed systems with certain trade-off have been developed for different languages in various studies. In daily life, however, casual and conventional ways are more commonly used in many languages, with varying inconsistency. Chinese (Mandarin) is a typical example of a language with a stable Romanization system, *pinyin*, which is used officially and in daily life. As to the case of Japanese, the *Hepburn Romanization* is the most common system for daily use, including passports, although certain variants in notation (e.g., around long vowels) are

allowed. Korean is a case with considerable variants in Romanization, where conventional or personalized spellings are prevalent.

In this study, we focus on the name Romanization of Khmer, a Southeast Asian language with limited NLP studies. Similar to the languages mentioned above, several Romanization systems have been designed for Khmer. However, the case of Khmer is most similar to that of Korean, where Romanization with conventional spellings is strongly preferable in practice. Compared with Korean, Khmer has a more complicated phonology, less clear syllable structures, and more etymologically oriented spellings, all of which make the task more problematic. Hence, statistical approaches based on real data are required to solve the problem. In this paper, we provide satisfactory solutions for the Khmer name Romanization task with detailed descriptions on data preparation, statistical approaches, and discussions. We believe this is the first comprehensive work for the specific task and we have released the data prepared and used in this study under a license of CC BY-NC-SA.¹

Specifically, we collect 7,658 Romanization instances from real Khmer person names. We then design an alignment scheme to manually annotate the Romanization instances. The scheme is carefully designed to realize a (1) consistent, (2) precise, and (3) monotonic alignment on grapheme level. In practice, the alignment is conducted manually at the very beginning, and automatic cross-checking with manual modification is conducted repeatedly to clean up mistakes. Finally, difficult and unusual instances are picked up and checked with further discussions between annotators. The elaborated manual alignment on a dataset with a considerable size thus provides a solid foundation for further investigation.

In the experiments, we first test rule-based approaches, which do not require training data but use manual rules. The performance is mediocre. Ad-hoc rules only provide insignificant gains. For statistical approaches, two standard machine learning methods, *conditional random fields* (CRF) and *support vector machine* (SVM), are tested. Because of the well-prepared grapheme-level alignment, the Romanization task is simplified to be a sequence labeling task. The performance is satisfactory: the precision on grapheme level reached .99 in cross-validation. We also conduct a direct sequence-to-sequence experiment without using manual alignment but applying a state-of-the-art bidirectional long short-term memory (LSTM) based recurrent neural network (RNN) approach. The performance is good, but does not match CRF's and SVM's results, which take advantage of manual alignment. Therefore, We consider that it is the high quality of our manually aligned data, rather than a sophisticated model, that contributes more to the task.

The remainder of the paper is organized as follows: In Sect. 2, we introduce the background of Khmer and related work on Romanization in NLP. In Sect. 3, we provide descriptions of the alignment scheme we designed and applied to annotate the data. Section 4 reports the evaluation and discussion based on experiments, and Sect. 5 provides a conclusion.

¹ <http://nptict.edu.kh/khmer-name-romanization-with-alignment-on-grapheme-level/>.

2 Background

Compared with the other Southeast Asian languages (e.g., Burmese and Thai), the phonology of Khmer has two obvious features. First, Khmer is **not** a tonal language, which is quite unusual in Southeast Asia. To compensate for the absence of tones, Khmer has a large set of vowel phonemes with a size of up to (at least) 31. Second, Khmer has abundant types of consonant clusters. A two-consonant cluster in a syllable’s onset is common (three at the maximum). Phonotactic constraints are loose in Khmer, where complex consonant clusters can appear at the beginning of a syllable. A complete list may contain up to around 80 types of consonant combinations.

The two phonemic features of Khmer are deeply related to Khmer’s abugida writing system. In a general abugida system, each standalone consonant letter can form a complete syllable with a hidden inherent vowel. The inherent vowel can be changed to other vowels or suppressed using various diacritic marks. The special feature of Khmer script is that there are two series (or registers) of consonant letters, which have different inherent vowels. The two series are usually mentioned as **a**-series (1st-series) and **o**-series (2nd-series). Furthermore, diacritics represent two vowels when added to corresponding consonant series,² which leads to a very complicated vowel system. Another feature of Khmer script is that the stacked consonants are very common. One reason is the abundant consonant clusters in phonology; another reason is the etymological spelling of Sanskrit (Pali) derived words. The stacked consonants are not strictly based on the syllable structure. Additionally (and more problematically), the *virama*, i.e., the diacritic used to suppress the inherent vowel, is absent in Khmer script,³ which makes the identification of onset and coda difficult.

Further introduction of Khmer’s phonology and script can be referred to in the following resources available on-line: a sketchy one [3] and a detailed one [6]. We provide examples in Figs. 1 and 2 for illustration. In the first example, three consonant letters, **2**, **4**, and **6** are stacked to a block to represent a consonant cluster and a dependent vowel diacritic **7** is attached to the block to form an unbreakable unit in writing. However, the first consonant **2** in this unit is the coda of the first syllable (**1 2**) and the left **4** and **6** are the onset of the second syllable (**4 6 7 8**). This is a typical instance of inconsistency between phonemic analysis and writing systems. Furthermore, the first and third units, both of which are composed of a single consonant letter (**1** and **8**, respectively), play different roles. The first unit stands for the onset and nucleus of the first syllable with the inherent vowel of letter **1**. The third unit is the coda of the second syllable, where the inherent vowel of letter **8** is suppressed, without any explicit notation in writing. The second example has a more consistent and

² Some diacritics always stand for one vowel. There are also diacritics serving as “shifter” to switch the series of certain consonant letters, which affect the vowel sound of other diacritics.

³ Actually, Khmer script has this diacritic, called *viriam*, but uses obsoletely.

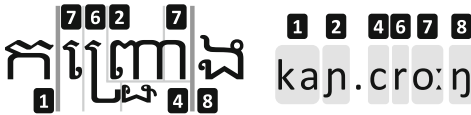


Fig. 1. Khmer word with the meaning of *fox*. The bold lines on the Khmer word show the boundary of the unbreakable unit in writing and the thin lines distinguish the components. The order of Khmer letters is marked by numbers. The missing **3** and **5** are invisible stacking operators for generating the unit of **2 4 6**. The **7** is a separated diacritic with two parts. IPA for the whole word and each letter is shown.

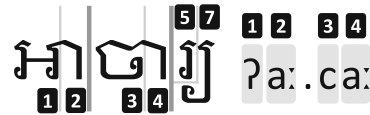


Fig. 2. Khmer word with the meaning of *teacher*, borrowed from Sanskrit *ācārya*. Notation in this figure is identical to that in Fig. 1. The missing **6** is the invisible stacking operator for generating the unit of **5 7**. The third unbreakable unit **5 7** is completely silent in pronouncing, but only written in orthography due to the etymology.



Fig. 3. A Khmer name composed of two words. The left side is the raw string pair and the right side is the grapheme-aligned pair. **4** is the space; **7** and **9** are stacking operators. The “.” stands for inserted inherent vowels on the Khmer side (annotated by a hedge) and for a silent placeholder on the Latin side.

clearer match between writing and pronouncing, in that the first and second units (**1 2** and **3 4**, respectively) exactly correspond to the two syllables. The problem here is the third unit (**5 6**), which is totally silent, but appears in writing only for the etymological reason. These examples illustrate the difficulties in even a pure phonemic transcription for Khmer script, where orthographic and etymological facts are involved. As for Romanization in practice, the task further suffers variants in mapping from phoneme to grapheme, as noted by the Latin alphabet.

In engineering practice, the Romanization task is a string-to-string transformation, which can be cast as a simplified translation task working on grapheme level rather than on word (or phrase) level with no (or few) reordering operations. Hence, general SMT techniques can be facilitated once training data are prepared. The phrase-based SMT plays a role of baseline in recent workshops [1,2], whereas neural network techniques provide further gains in performance [4,5]. Although neural network-based, pure string-to-string approaches prove powerful on different transliteration tasks, there is still room for improvement,

especially on tasks between different writing systems. For example, the Thai-to-English task, which is similar to our task, has relatively poor performance in NEWS 2015.⁴ The problem around diacritics in abugida is also stated in [7]. General techniques may offer acceptable solutions overall, but specialized investigation and processing are required for further improvement on tasks for specific languages, or language pairs.

3 Data

3.1 Collection

We collect 7,658 real Khmer names with corresponding Romanization in pairs. Khmer names usually consist of two words, a family name coming first, followed by a given name, separated by a space in writing. However, the family name is not an obliged element, i.e., a person may have no family name but only a given name. On the other hand, a given name may contain more than one word. As a result, for a name containing two words, it is not always clear that it is a family-given name pair or a two-word given name. We do not explicitly distinguish the family name and given name in our data. The spaces in names are treated as an ordinary character and kept constant in Romanization. That is, Romanization is always word-by-word, without any merging or splitting operations. An example of a two-word Khmer name with its Romanization is illustrated on the left side of Fig. 3.

3.2 Overall Principles in Alignment

As mentioned, phonemic syllables and unbreakable writing units do not match well in Khmer script. Consistent alignment principles are thus difficult to establish if the syllables or units are taken as atoms in processing. Furthermore, there are numerous types of syllables and writing units that may contain up to four phonemes, which are too complex for statistical model learning because of sparseness. Based on these facts, we prefer a pure character-level alignment, where standalone consonant letters, diacritics, and the invisible staking operator are separated and treated equally as grapheme on the Khmer side.

A consequent problem is the inherent vowels, once we completely ignore the syllable structure in a character-based alignment. We cannot judge whether a standalone consonant stands for only one consonant or contains a further inherent vowel, due to ambiguity in the writing system. We thus apply a scheme to insert a mark to represent the inherent vowel for all the “bare” consonant letters (i.e., consonant letters without any diacritics or stacking operators) to establish a consistent alignment. This insertion is thus decisive based on the surface spelling, and the ambiguity on the presence or absence of the inherent

⁴ The original names are western names in the Thai-to-English task, which may make the grapheme correspondence more varied and inconsistent.

vowel is converted to whether the inserted mark is silent.⁵ As a result, the problem is treated as uniformly as the other silent characters.

The right side of Fig. 3 illustrates an alignment example. The consonant letters here are **1**, **3**, **5**, **6**, **8**, and **10**, where **3** and **5** are bare consonant letters,⁶ after which the inherent vowel is inserted (noted by a dot here and indicated by a wedge without original index). Then a character-by-character alignment can be established.⁷ According to the overall principles, Khmer consonant letters are aligned to Latin consonant graphemes; diacritics, including inserted inherent vowels, are aligned to Latin vowel graphemes; and any silent part is aligned to a placeholder.⁸ Further alignment details caused by specific use is described in the following subsection.

In Fig. 3 as well as in our practice, we use the same mark for the Khmer side inserted inherent vowel and the silent placeholder on the Romanization side. This introduces no confusion because the mark is decisively inserted on the Khmer side and decisively deleted in the Romanization results.

3.3 Special Cases in Alignment

The alignment under our overall principles is almost monotonic and one-to-one. However, there are two specific cases to be discussed: (1) multiple diacritics for one consonant letter and (2) stacked consonants letters for a single phoneme.

The first case is mainly caused by the *anusvara* (adding nasal ending), *visarga* (adding aspiration ending), and *series shifters* changing the a-/o-series of consonant letters. Figure 4 illustrates the examples. When a consonant letter takes an anusvara directly, it is not bare anymore, so the Latin letters for inherent vowel and the nasal ending (commonly transcribed into M) are taken as one grapheme aligned to the anusvara. The anusvara can, however, modify other diacritics to add a nasal ending, where only the M endings are aligned to anusvara and the vowel value is taken by the preceding diacritics. The first two instances in Fig. 4 show the difference in alignment on anusvara. The alignment around visarga, which is commonly transcribed into S, is identical to that around anusvara. The middle two instances in Fig. 4 show the alignment around the series shifter, which is similar to the anusvara (*visarga*), but inserted between a consonant letter and other diacritics. The shifter is actually modifying the preceding consonant letter, so the value of the vowel will be afforded by other diacritics if there is any. The shifter itself has no specific value unless it is the only diacritic of a consonant.

⁵ Generally, the diacritics in an abugida system are observed as vowel shifters to change the inherent vowel. From the viewpoint of this study, the diacritics are actually treated as vowel notes and the inherent vowel is treated as one with a zero-alternant form, which the insertion processing makes explicit.

⁶ **2** and **11** are diacritics for **1** and **10**, respectively, and **6** and **8** are followed by staking operators. So these four consonant letters are not bare.

⁷ As the task is to transform Khmer script to Latin script, the graphemes are not guaranteed to be single letters on the Romanization side, e.g., the **5** corresponds to CH here.

⁸ The stacking operator is always aligned to a silent placeholder.

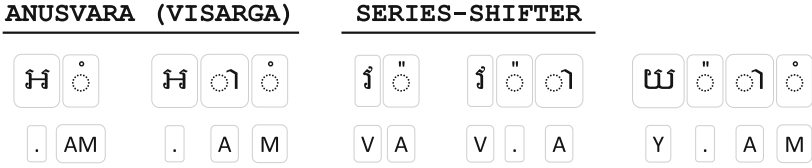


Fig. 4. Alignment principles around anusvara (visarga), series shifters, and an alignment example for a Chinese-style name with both diacritics.



Fig. 5. Alignment examples of two western-style names, where the consonants Z and F are non-native phonemes of Khmer. The original value (common transcription) of consonant letters combined to represent the two sounds is placed upward.

The rightmost example in Fig. 4 is a real name taking both anusvara and the shifter simultaneously, aligned according to our principles.

The second case is caused by certain non-native consonants in Khmer, mainly appearing in loanwords, and here, appearing in westernized names. Figure 5 provides two real examples. It can be observed that Z is represented by stacked Khmer letters for H and S, and F is represented by stacked Khmer letters for H and V. The consonant letter of H is commonly stacked over another consonant letter of an approximate pronunciation to represent a non-native consonant. Therefore, the corresponding Latin consonant graphemes are aligned to the second Khmer letters and the first Khmer letter H is aligned to a silent placeholder. In Fig. 5 (on the right), the stacked consonants for F are further modified by two diacritics, where the first is a series shifter.

As illustrated in this subsection, the combination of consonant letters and various diacritics is complicated in Khmer script, and the grapheme-level alignment we designed can cover different cases consistently, to offer a monotonic and one-to-one alignment. However, the Romanization of Khmer characters is not totally position-free under our alignment scheme, and at least a window of tri-gram on Khmer characters is required to provide enough information for a correct Romanization.

4 Experiment

4.1 Questions and Approaches

The well-aligned data prepared in this study have already provided a solid foundation of the Romanization task. Therefore, we investigate the following two

questions through experiments: (1) to what extent can statistical approaches help the task and (2) to what extent can our manual alignment help statistical approaches.

For the first question, we first try a rule-based transcription to investigate the performance. The rules used are edited manually, based on conventional spellings in Romanization rather than academic standards. Then we try a state-of-the-art neural network-based sequence-to-sequence approach trained on surface string pairs in our data, without using the alignment. The neural network outperforms the rule-based approach by a large margin. The results illustrate the necessity of real data and statistical approaches in this task.

For the second question, we further try two standard and widely used machine learning approaches, CRF and SVM, trained on the grapheme-level alignment. Because the alignment has cleaned up two difficult problems in transformation, i.e., source-side segmentation (actually, character-by-character) and source-to-target alignment, the task is converted to a sequence labeling task, to which these compact and efficient approaches are applicable. The results yielded by the two approaches outperform the neural network’s results, demonstrating that the manual alignment does provide useful information and boosts the performance efficiently.

The details of the experiments are described in the following subsections. Experimental results are evaluated in two ways: the accuracy on source-to-target grapheme transcription (GRAPH) and the accuracy on target strings (LATIN). GRAPH is based on grapheme-level alignment, where all Khmer letters, including the inserted inherent vowel, are counted in the calculation.⁹ GRAPH cannot be applied to the sequence-to-sequence experiments as the alignment is not an explicit variable in the processing and final results. For LATIN, we simply apply the BLEU score [11], the most common measure in SMT, on alphabet level, where silent placeholders in Romanization are deleted before calculation.¹⁰

4.2 Rule-Based Transcription

Although the Romanization of Khmer is not consistent, there is preference for each letter. Figures 6 and 7 show the most preferred transcription of Khmer letters. A naïve mapping based on the listed transcriptions cannot reach a satisfactory performance. The accuracy is .746 on GRAPH and .515 on LATIN. When we focus more on some conventional spelling features, so as to double the consonant letters after short vowels or to transform specific stacking consonants, (e.g., the case in Fig. 5), we have a better result, a GRAPH of .909 and a LATIN of .821. Further improvement is difficult. We try ad-hoc rules in an exhaustive way and find the upper boundary of the rule-based mapping to be around .95 for

⁹ Spaces are not counted because they are always maintained constant. Stacking operators as well as other silent Khmer letters are counted.

¹⁰ Spaces are taken as one character in the BLEU calculation.

GRAPH and .88 for LATIN. Although the performance is not perfect, we consider it reasonable and acceptable in terms of simplicity.¹¹

4.3 Sequence-to-Sequence Transliteration

In recent research, direct sequence-to-sequence approaches launched by neural network techniques have been widely applied in various NLP tasks. We experiment a state-of-the-art LSTM-based RNN approach with a bidirectional search in decoding¹² [9]. The approach performs well on different transliteration tasks.

The experiment is conducted using an eight-fold cross-validation on our data without using manual alignment. The entire data are split into eight parts, with each part taken as a test set and the left parts used for training. As separated development data are needed for tuning the iteration times, one-thirtieth of the training data is sampled. Other hyper-parameters are based according to the original paper: embedding size is 500, hidden unit dimension is 500, and batch size is 4. AdaDelta is used for optimization with a decay rate ρ of 0.95 and an ϵ of 10^{-6} .

The result of LSTM-based RNN reaches .953 in terms of LATIN. Compared with the rule-based mapping result, a machine learning approach clearly performs better on the task, even without applying any specific *a priori* knowledge. The comparison of the two approaches provides a clear and solid answer to the first question.

4.4 Alignment-Based Sequence Labeling

We test two standard machine learning approaches, CRF and SVM, by handling the task in a sequence labeling manner. That is, the Latin graphemes (including the silent placeholder) are tried as labels for each Khmer character.

We use the **CRF++** toolkit¹³ [8, 12] and the **KyTea** toolkit¹⁴ [10] for CRF and SVM experiments, respectively. Both toolkits are open-sourced. Similar to RNN experiments, CRF and SVM experiments are also cross-validated, where the eight-, four-, and two-fold results are tested for comparison. As to the settings of the two approaches, we basically use up to tri-gram on Khmer characters as input features. As mentioned, this is the minimum size of a window to provide adequate information under our alignment scheme. Specifically, the `-charn` and the `-charw` options are set to three for **KyTea**. The features used for **CRF++** are C_n^{n+k} ($k \in [1, 2]$, $n \in [-k, 0]$) for character sequences and C_n ($n \in [-2, 2]$) for single characters. As the training speed of **KyTea** is very fast under this setting, we ultimately try an exhaustive leave-one-out experiment, in which each instance is left for testing and all the rest instances are used for training. Table 1 lists the evaluation on the whole dataset.

¹¹ A Python implementation for the rule-based transcription with different layers of rules is available at <http://www2.nict.go.jp/astrec-att/member/mutiyama/software.html>.

¹² An open-sourced tool is available at <https://github.com/lemaoliu/Agtarbidir>.

¹³ <http://taku910.github.io/crfpp/>.

¹⁴ <http://www.phontron.com/kytea/>.



Fig. 6. Preferable transcription for Khmer consonant letters. A-series and o-series letters are listed. The gray letters are rarely used and never appear in our data. The final diacritics in the two series are the two series shifters, which change the consonant letters in the other series to their own series.

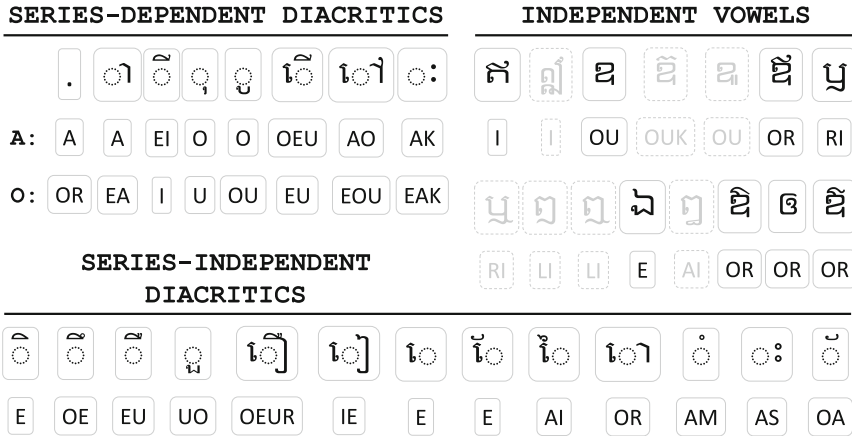


Fig. 7. Preferable transcription for different Khmer vowel diacritics and letters. Diacritics with consonant series-dependent transcriptions are listed on the upper left, a-series, and o-series respectively. Diacritics with identical transcriptions are listed at the bottom. These diacritics may stand for different vowels for consonants from different series, without being reflected in Romanization. In the upper right, standalone letters for vowels are listed. Rare vowels not appearing in our data are marked in gray.

Table 1. Evaluation results (GRAPH/LATIN) of CRF and SVM in cross-validation.

	2-fold	4-fold	8-fold	leave-1-out
CRF	.987/.974	.988/.976	.989/.977	—
SVM	.988/.977	.989/.978	.990/.979	.990/.980

The performance of the two approaches is nearly identical. The performance on GRAPH is around .99 and on LATIN is around .98, which outperforms the RNN's .95. The results is satisfactory and reasonable. As to the answer to the second question, we conclude that the Khmer Romanization task does not require features in a long distance, which RNN can model well, but precise local alignment provides useful and efficient information contributing to the performance.

As to engineering issues in practice, it takes **hours** to train an RNN model on more than 6,000 instances in eight-fold cross-validation, while to train the SVM model in **KyTea** only takes **seconds**. Therefore, we consider RNN to be a superfluous approach for the Khmer Romanization task. As Table 1 shows, two-, four-, eight-fold, and leave-one-out results actually do not differ much, which indicates that several thousand instances are sufficient for the model training.

5 Conclusion

In this paper, we focus on the task of Khmer name Romanization, a task not huge, but with its own difficulties. By collecting and elaborately aligning more than 7,000 real instances, we provide a solid foundation for the task. Experimental results demonstrate that a rule-based approach is not sufficient to solve the Romanization task, while statistical machine learning approaches trained on our manual alignment can achieve an accuracy of .99 on the grapheme level. Therefore, we believe that the Khmer name Romanization task has actually been solved, provided our data and standard machine learning techniques. As our manually aligned dataset plays a key role in solving the task, we release it under a license of CC BY-NC-SA for the research community.

References

1. Banchs, R.E., Zhang, M., Duan, X., Li, H., Kumaran, A.: Report of NEWS 2015 machine transliteration shared task. In: Proceedings of NEWS, pp. 10–23 (2015)
2. Costa-jussà, M.R.: Moses-based official baseline for NEWS 2016. In: Proceedings of NEWS, pp. 88–90 (2016)
3. Ehrman, M.E., Sos, K., Kheang, L.H.: Contemporary Cambodian – grammatical sketch (1974). <https://www.livelingua.com/fsi/Fsi-ContemporaryCambodian-GrammaticalSketch.pdf>
4. Finch, A., Liu, L., Wang, X., Sumita, E.: Neural network transduction models in transliteration generation. In: Proceedings of NEWS, pp. 61–66 (2015)
5. Finch, A., Liu, L., Wang, X., Sumita, E.: Target-bidirectional neural models for machine transliteration. In: Proceedings of NEWS, pp. 78–82 (2016)
6. Huffman, F.E.: Cambodian system of writing and beginning reader with drills and glossary (1970). <http://www.pratyeka.org/csw/hlp-csw.pdf>
7. Kunchukuttan, A., Bhattacharyya, P.: Data representation methods and use of mined corpora for Indian language transliteration. In: Proceedings of NEWS, pp. 78–82 (2015)
8. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: Proceedings of ICML, pp. 282–289 (2001)

9. Liu, L., Finch, A., Utiyama, M., Sumita, E.: Agreement on target-bidirectional LSTMs for sequence-to-sequence learning. In: Proceedings of AAAI, pp. 2630–2637 (2016)
10. Neubig, G., Nakata, Y., Mori, S.: Pointwise prediction for robust, adaptable Japanese morphological analysis. In: Proceedings of ACL-HLT, pp. 529–533 (2011)
11. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: BLEU: a method for automatic evaluation of machine translation. In: Proceedings of ACL, pp. 311–318 (2002)
12. Sha, F., Pereira, F.: Shallow parsing with conditional random fields. In: Proceedings of HLT-NAACL, pp. 134–141 (2003)



Burmese (Myanmar) Name Romanization: A Sub-syllabic Segmentation Scheme for Statistical Solutions

Chenchen Ding^{1(✉)}, Win Pa Pa², Masao Utiyama¹, and Eiichiro Sumita¹

¹ Advanced Translation Technology Laboratory, ASTREC,
National Institute of Information and Communications Technology,
3-5 Hikaridai, Seikacho, Sorakugun, Kyoto 619-0289, Japan
{chenchen.ding,mutiyamam,eiichiro.sumita}@nict.go.jp

² Natural Language Processing Lab, University of Computer Studies,
Yangon, Myanmar
winpapa@ucsy.edu.mm

Abstract. We focus on Burmese name Romanization, a critical task in the translation of Burmese into languages using Latin script. As Burmese is under researched and not well resourced, we collected and manually annotated 2,335 Romanization instances to enable statistical approaches. The annotation includes string segmentation and alignment between Burmese and Latin scripts. Although previous studies regard syllables as unbreakable units when processing Burmese, in this study, Burmese strings are segmented into well-designed sub-syllabic units to achieve precise and consistent alignment with Latin script. The experiments show that sub-syllabic units are better units than syllables for statistical approaches in Burmese name Romanization. The annotated data and segmentation program have been released under a CC BY-NC-SA license.

1 Introduction

Linguistically, Romanization is the task of transforming a non-Latin writing system into Latin script. It is a language-specific task in natural language processing (NLP), and an important module in a statistical machine translation (SMT) system, required to handle the unknown proper nouns that occur when translating a language using non-Latin script into languages using Latin script.

The task of Burmese (Myanmar) name Romanization is investigated in this study. Burmese is an understudied language and there are not many resources available for its study, although research on Burmese begun in recent years. Current research on Burmese focuses on applying available techniques to perform basic tasks in NLP such as tokenization [5], parsing [4], and SMT [3], as well as to perform basic tasks in speech processing, such as automatic speech recognition [10] and speech synthesis [17].

This study thus proceeds from raw Romanization instance collection to manual annotation and finally empirical investigation. The annotation was conducted

considering Burmese phonology and phonotactics, as well as the conventional Romanization spellings. Statistically based experiments were then performed on the annotated data. The contribution of this study is that well-designed sub-syllabic units rather than integrated syllables in Burmese, are regarded as segments and proved to be efficient representations through experimental results. Compared with syllables, these sub-syllabic units, although they need a specific process to extract, reduce the number of segment types, which can significantly reduce the sparseness for statistical approaches. The annotated data and a `python` implementation of the sub-syllabic segmentation scheme have been released for NLP community under a `CC BY-NC-SA` license.¹

2 Related Work

The Romanization task is a string-to-string transformation that can be cast as a simplified translation task working on grapheme level rather than word (or phrase) level with no (or few) reordering operations. Thus, general SMT techniques can be facilitated once training data are available. Phrase-based SMT plays an important role in Romanization tasks and has been taken as a baseline system in recent workshops [1, 2], whereas recent neural network techniques provide further gains in performance [6, 7].

Although there are several available transcription guidelines for Burmese Romanization,² conventional and inconsistent spellings are more prevalent in practice use. To the best of our knowledge, there is still no systematic research on this specific topic in NLP. Related studies may include the grapheme-to-phoneme (G2P) study on Burmese [16], which is a task for speech processing rather than NLP, and thus relatively large units in utterance, i.e., syllables, were used in this research. Compared with G2P, the Romanization task further suffers from the diversity of conventional spellings. For example, a common Burmese rhyme `/-in̩/` may have four equal variants `-in`, `-inn`, `-yn`, and `-ynn` in practice. Non-phonological spellings are also common, such as representation of a creaky tone by an ending `t` or using English-like spellings for similar phonemes, e.g., `-ike` for `/-ai̯/`, and `-oo` for `/-u̯/`.

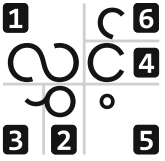
As a primary contribution of this study, we provide a solid basis for the task of Burmese name Romanization, for the first time, that includes all components from data to practical techniques. We use sub-syllabic units, specifically, onset, rhyme, and tone, and experimentally prove that they are more flexible and precise units than syllables for statistical approaches. In a wider sense, as the study is based on an insightful consideration of Burmese phonology and phonotactics, we believe the sub-syllabic units are also applicable for other related Burmese processing tasks.

¹ http://www.nlpresearch-ucsy.edu.mm/NLP_UCSY/name-db.html.

² Typical ones are the Myanmar Language Commission Transcription System, the Library of Congress' ALA-LC Romanization index system for Burmese (<http://www.loc.gov/catdir/cpso/romanization/burmese.pdf>), and the Okell's system [13].

3 Burmese Syllable

3.1 Syllable Composition

 Burmese applies an abugida writing system, i.e., consonant letters stand for a syllable with an implicit inherent vowel, and diacritics modify consonant letters to change/depress the inherent vowel, to form consonant clusters, or to change tones. A Burmese syllable is illustrated in the beginning of the section, where the numbers indicate the order of the characters in the composition. **1** and **4** are consonant letters, and **2**, **3**, **5**, and **6** are diacritics. **2** and **3** form consonant clusters with **1**. **5** is a tone mark. **6** is the virama to depress the inherent vowel of **4** to form the coda of the syllable. As illustrated, Burmese syllables are usually composed of multiple characters, while syllables can be identified by rules, i.e., all diacritics and consonant letters with a virama (inherent vowel depressor) must be attached to the letter they modify [5].

3.2 Syllable-Based Processing and Limitation

Syllables are usually treated as basic atoms, i.e., unbreakable units, in related NLP studies such as Burmese word segmentation [5] or SMT [3]. In these studies, the syllable identification is the first step before any further processing. This treatment is proper regarding to the nature of those morphology or syntax related NLP tasks, as structures and fragments within syllables generally have no significant meaning attributing to sentence-level analysis.

Syllable-based processing is also a straightforward way in the Romanization task. However, the structure within syllables have a clearer and more important role in the task, because Romanization is a task more related to phonological and phonotactic features. On the other hand, the syllable turns to be a too integrated unit for detailed analysis and modeling in terms of phonology, where different parts within a syllable should be reviewed. To solve the Romanization task more efficiently, we step into the syllable structure and extract sub-syllabic segments, to achieve a precise and consistent correspondence between Burmese and Latin scripts by establishing a monotonic and one-to-one alignment.

An intuitive example on the proposed sub-syllabic segmentation is illustrated in Fig. 1, with the comparison of syllables and characters. As illustrated, syllables are integrated units but with relatively complex internal structures, i.e., syllables are too large segments in processing; character-level analysis, oppositely, turns too detailed that the correspondence between the two writing systems are difficult to sort out, i.e., characters are too tiny segments. In practice, the syllable-based processing may cause serious sparseness in model learning due to the large amount of different types, and the character-based processing introducing excessive and tricky alignment. The proposed sub-syllabic segmentation is thus at a good balance point of integrity and preciseness between the two extremes in handling the Romanization task.

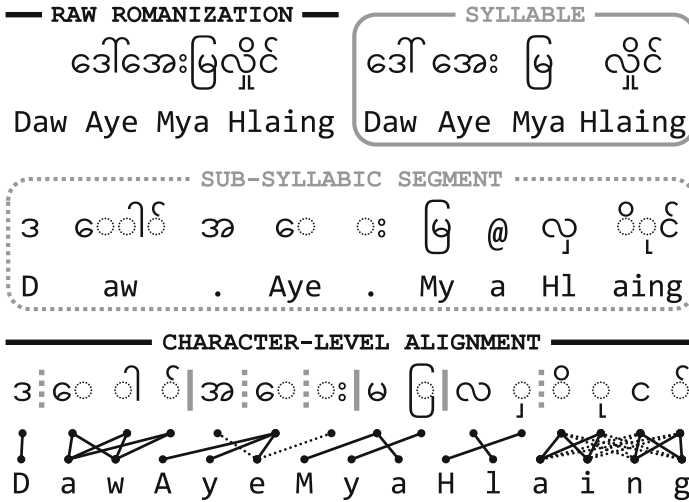


Fig. 1. Overview of sub-syllabic segmentation. Upper-left is a raw Romanization instance; upper-right shows syllables. In the sub-syllabic segmentation, @/. show inserted/silent segments. The character-level alignment is complex, where dash lines show the spellings affected by surrounding Burmese characters. Boundaries of syllables and sub-syllabic units are illustrated by solid and dash vertical bars, resp., in the character-based analysis.

As syllables are integrated units in the Burmese script, the segmentation scheme requires further transposition and insertion processing. The descriptions of segmentation and the transposition/insertion processing are given in the following subsections respectively.

4 Sub-syllabic Segmentation

4.1 Onset-Rhyme-Tone Segmentation

Figure 2 is a diagram of the overall sub-syllabic segmentation scheme, where the three gray blocks are the sub-syllabic units designed in this study. It is relatively intuitive to divide a Burmese syllable into two segments, onset and rhyme, despite the difficulties introduced by medial consonants, where multigraphs of Latin letters may be used for Burmese letter clusters (Fig. 3). The rhyme is not further dividable except to stripe tones annotated by explicit marks.

Specifically, four diacritics are used to represent the medial consonants, i.e., *yapin*, *yayit*, *wahswe*, and *hahto* to represent /-j-/, /-j-/,³ /-w-/, and /-h-/,⁴ respectively. As shown in Fig. 2, *yapin*, *yayit*, and *hahto* are placed into the onset

³ *Yayit* originally represents /-r-/ while in the modern standard Burmese the phoneme /r/ has been merged into /j/.

⁴ Actually a voiceless sign, e.g., changing /n-/ to /ŋ-/.

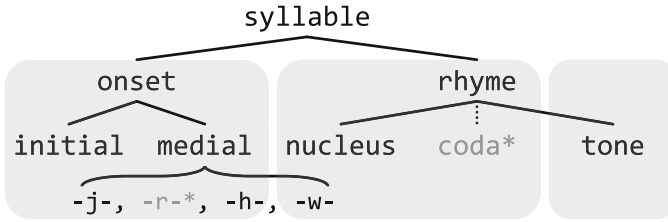


Fig. 2. Proposed sub-syllabic segmentation.

segment, but *wahswe* into the rhyme segment. This scheme is adopted for two reasons. (1) The phonotactical constraints are strict on *yapin*, *yayit*, and *hahto*, but loose on *wahswe*. *Hahto* is only used on sonorants (nasals and approximants) and *yapin/yayit* only on velars and bilabials.⁵ In contrast, *wahswe* can be freely combined with all consonant phonemes with the trivial exception of /w-/.⁶ (2) *Yapin*, *yayit*, and *hahto* may change the property of the initial consonants while *wahswe* may change the property of the nucleus vowels. Besides the voiceless marker of *hahto*, *yapin/yayit* palatalize velar consonants.⁷ Contrarily, *wahswe* may add a rounded feature to the following nucleus vowels.⁸ These two issues affect conventional Romanization spelling. As shown in Fig. 3, multigraphs of Latin letters are used to transcribe onset clusters with *yapin*, *yayit*, and *hahto*, or *wahswe*-rhyme clusters. Consequently, our scheme is the only feasible way to segment Burmese onset and rhyme to achieve a monotonic and one-to-one alignment with Latin script in conventional Romanization.

Burmese has two codas, i.e., nasal and glottal, and three tones, i.e., creaky, low, and high. The three tones can be combined freely for open and nasal-ended syllables, but not for glottal ended ones. The glottal ending thus may be regarded as a fourth tone in some analysis. Further, the nasal ending can be regarded as a nasalization of nucleus vowels. Hence, the coda is not a necessary component from an extreme viewpoint, which places the nasal ending into nuclei and the glottal into tones. However, in the writing system, the nasal and glottal endings are represented by consonant letters with a virama. These “coda-letters” affect nucleus vowels, so that they are not completely detachable. As a result, only two tone marks, i.e., the visarga (high) and *aukmyit* (creaky) are segmented in our scheme, as they are “pure” tone marks,⁹ and any further segmentation would ruin the integration of the rhyme.

⁵ *Yapin* can also be combined with /l-/.

⁶ /ʔw-/ may be argued in some references. The combination appears marginally in borrowing words and interjections.

⁷ E.g., /kj-/ is actually /c-/ or /tʃ-/.

⁸ E.g., changing /-aʔ/ to /-ʊʔ/ and changing /-an/ to /-ʊn/.

⁹ The visarga is usually not transcribed and *aukmyit* is inconsistently represented by a final **t** in Romanization.

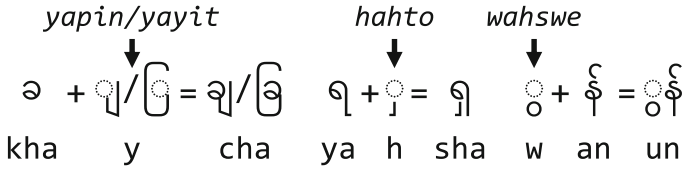


Fig. 3. Examples of common Latin multigraphs used for Burmese letter clusters.

4.2 Transposition and Insertion

Although the sub-syllable units have been figured out in the previous subsections, a raw Burmese string requires pre-processing for the segmentation, due to the coding order of Burmese script. Examples of segmentation with transposition and insertion are shown in Figs. 4 and 5, respectively.

For the onset-rhyme segmentation, an onset segment with multiple components is coded in the order of “*C + yapin/yayit + wahswe + hahto*,” where *C* is the initial consonant.¹⁰ It is obvious that the segmentation cannot be conducted under the coding order once *wahswe* and *hahto* appear simultaneously. Hence, a **wahswe-hahto swapping** is required. For the rhyme-tone segmentation, one problem is the order of the *aukmyit* and virama in nasal-ended creaky-toned syllables.¹¹ As the standard order should be “*aukmyit + virama*” in coding,¹² an **aukmyit-virama swapping** is required.

In addition to the transposition pre-processing, a “dummy rhyme” is inserted for all the “bare onsets”, i.e., syllables without an explicit rhyme where the implicit inherent vowel performs the role of the nucleus. This insertion post-processing can provide a precise alignment between Burmese and Latin scripts, as the vowels are always explicit in an alphabetic system but may be implicit in an abigida system. Note that the example in Fig. 4 and the final example (*mwa*) in Fig. 5 do not require this insertion because the rhyme segments are not empty, i.e., “coda-letter” and *wahswe* can perform the role of the rhyme segment.

5 Annotated Data

Our Burmese Romanization instances were first collected from students and faculty names in a university. More instances were then collected from the Internet, including names of public figures and names from minority nationalities in Myanmar.

To process the raw instances, we first segmented Burmese and Latin strings roughly into consonant and vowel segments and used a greedy algorithm to

¹⁰ Multiple medial consonants for one initial consonant is possible while *yapin* and *yayit* cannot appear simultaneously.

¹¹ As mentioned, glottal endings take no tones.

¹² However, the swapped order may introduce no problem in displaying, so both orders are used in daily typing.

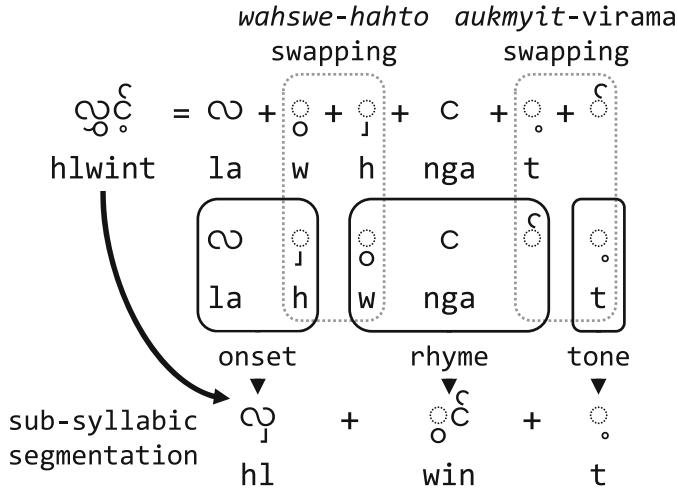


Fig. 4. Transposition before segmentation.

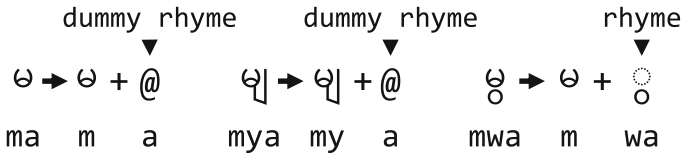


Fig. 5. Insertion for onsets without a rhyme.

generate a preliminary alignment with the help of an aligner.¹³ Around 60% of the instances were aligned consistently in this stage, and we manually checked the errors to refine the overall segmentation scheme. In the second round, we filtered out questionable instances by cross-checking. Errors were found and cleaned in around 10% of the instances. In the third round, more tolerable instances, i.e., Burmese names with more than one acceptable Romanization, were added. We obtained a final total of 2,335 annotated instances.

Specific annotation samples are illustrated in the appendix attached to this paper.

6 Experiment

In recent research, direct sequence-to-sequence approaches launched by neural network techniques have been widely applied in various NLP tasks. We experiment a state-of-the-art long short-term memory (LSTM) based recurrent neural network (RNN) approach with a bidirectional search in decoding¹⁴ [9]. The approach performs well on different transliteration tasks. We also test two standard

¹³ Using **GIZA++** [12] at <http://www.statmt.org/moses/giza/GIZA++.html>.

¹⁴ An open-sourced tool is available at <https://github.com/lemaoliu/Agtarbidir>.

machine learning approaches, conditional random fields (CRF) and support vector machine (SVM), by handling the task in a sequence labeling manner. We use the **CRF++** toolkit¹⁵ [8, 15] and the **KyTea** toolkit¹⁶ [11] for CRF and SVM experiments, respectively.

All the experiments were cross-validated. The RNN handles the task in a sequence-to-sequence way on character-level, without using any *a priori* knowledge,¹⁷ while CRF/SVM take advantage of the designed segmentation and manual alignment. The features for CRF/SVM were segments up to tri-grams. Specifically, the `-charn` and the `-charw` options are set to three for **KyTea**. The features used for **CRF++** are C_n^{n+k} ($k \in [1, 2], n \in [-k, 0]$) for segment sequences and C_n ($n \in [-2, 2]$) for single segments. The settings from the original paper were used for the RNN: embedding size is 500, hidden unit dimension is 500, and batch size is 4. **AdaDelta** is used for optimization with a decay rate ρ of 0.95 and an ϵ of 10^{-6} . We evaluated the experimental results using two metrics: the accuracy of target segments (SEG),¹⁸ and the accuracy of target strings (STR) where the BLEU score [14] at Latin letter level was applied.

The generalized LSTM-based RNN approach cannot handle the task well on our data. The STR is around .72 in 8-fold cross validation. The RNN actually generates “Burmese-styled” Latin transcriptions but inaccurate. We thus consider the performance to be reasonable and attribute the causes to (1) the data size, which is insufficient to support an RNN model, and (2) the intrinsically complex mapping between the two writing systems, where character-level many-to-many alignment is common and is difficult to model.

The CRF/SVM results using sub-syllabic units are listed in Table 1. The performance of the two approaches are similar: SEG is around .95 and STR is around .91. These results are acceptable, but there is still room for improvement. For comparison, Table 2 shows the results using syllables.¹⁹ As the segments are different, the SEG of the two tables cannot be compared directly. However, STR in Table 2 is lower than in Table 1, with statistical significance at the $p < 1\%$ level. The main cause of this difference is the sparseness, which is evidenced by the percentage of unknown segments (UNK) in the tables. Specifically, there are 548 types of syllables in our data, while the sub-syllabic segmentation reduces the number of types to 136, which largely reduces the sparseness. Consequently, we conclude that the sub-syllabic units designed in this study provide an efficient interface for machine learning approaches to handle the Romanization task.

¹⁵ <http://taku910.github.io/crfpp/>.

¹⁶ <http://www.phontron.com/kytea/>.

¹⁷ I.e., on the level in the bottom rank in Fig. 1, with no explicit alignment or unit boundaries between characters.

¹⁸ SEG cannot be applied to the RNN approach as the alignment and segmentation are not explicit variables.

¹⁹ I.e., the results in Tables 1 and 2 are based on the middle and upper-right parts in Fig. 1, respectively.

Table 1. CRF/SVM results on sub-syllabic units.

	2-fold	4-fold	8-fold
SEG	.946/.944	.948/.947	.947/.947
STR	.912/.907	.913/.911	.913/.910
UNK	0.13%	0.10%	0.09%

Table 2. CRF/SVM results on syllables.

	2-fold	4-fold	8-fold
SEG	.877/.882	.886/.888	.887/.889
STR	.878/.887	.888/.893	.890/.894
UNK	3.01%	2.48%	2.31%

7 Conclusion and Future Work

In this study, we focused on the Burmese name Romanization task, from the preparation of deliberately segmented and aligned data to experiments and discussion of statistical approaches. We are collecting more Burmese Romanization instances including the names of places and organizations. The annotated data and segmentation program have been released under a CC BY-NC-SA license to promote the research of NLP on Burmese.

Appendix

Figure 6 shows specific annotation instances for a further illustration and demonstration. The data are organized in a three-section format of

- original Burmese name,
- original Romanization, and
- aligned Burmese/Latin graphemes,

separated by |||.

The descriptions of specific instances are as follows.

- I. An ordinary Romanization instance.
- II. A Burmese name with a western expression (**Grace**) as a component. Generally, such western expressions are segmented according to the Burmese spellings. In this instance, **Grace** is segmented into /G /@ /r /a /@ /ce. Notice that we just apply the same @ for the dummy vowel on Burmese side and for the silent placeholder on Latin side, which causes no confusion.
- III. A Burmese name derived from Pali (**Wanna**),²⁰ where stacked consonants appear (/n /n). The stacked consonants are split and aligned to separate

²⁰ The Romanization instance is directly taken from the released data set. A more common Romanization of the Pali-derived name is **Wunna**.

- I. ဒေါ်ဝင်းပေ ||| Daw Win Pa Pa ||| ဒ/D ဝေါ်/aw ဝ/W င်/in ဝး/ဝဲ ဝ/P ဇ/a ဝ/P ဇ/a
- II. ဂရိတ်နွမ်း ||| Grace Nuam ||| ဂ/G ဇ/ဝဲ ရ/r ဝဲ/a ဝဲ/ဝဲ စ်/ce န်/N ဝဲ/မံ/uaam ဝး/ဝဲ
- III. မောင်ဝဏ္ဏ ||| Maung Wanna ||| မ/M ဝောင်/aung ဝ/W ဇ/a ထာ/n ဝဲ/ဝဲ ထာ/n ဇ/a
- IV. ရာသေကြိန် ||| Yar Za Thingyan ||| ရ/Y ထာ/ar ဇ/Z ဇ/a သ/Th င်/in ဝဲ/ဝဲ ကြ/gy န်/an
- V. နော်သီဂီထွန်း ||| Naw Theingi Htun ||| န်/N ဝောင်/aw သ/Th ဝိုင်/ein ဝဲ/ဝဲ ဂ/g ဝီ/i ထာ/Ht ဝဲ/နံ/un ဝး/ဝဲ
- VI. ဒေါ်သညာဝင်း ||| Daw Thinzar Win ||| ဒ/D ဝေါ်/aw သ/Th ဇ/i ည်/n ဝဲ/ဝဲ ဇ/z ထာ/ar ဝ/W င်/in ဝး/ဝဲ
- VII. ဒေါ်သညာဝင်း ||| Daw Thin Zar Win ||| ဒ/D ဝေါ်/aw သ/Th ဇ/i ည်/n ဝဲ/ဝဲ ဇ/z ထာ/ar ဝ/W င်/in ဝး/ဝဲ

Fig. 6. Specific annotation instances on Burmese name Romanization.

Latin letters. If no doubled Latin letters are used, the second Burmese character will be simply aligned to a silent placeholder \emptyset . The stacking operator is always aligned to \emptyset .

- IV. A Burmese name with complex stacking, that the rhyme of the previous syllable ($/in$) is stacked with the following onset ($/gy$).
- V. A Burmese name with more complex stacking, that part of the rhyme of the previous syllable ($/ein$) is stacked with the following onset ($/g$), which is taking a further vowel diacritic ($/i$). The instances IV. and V. illustrate the necessity on the segmentation of stacked characters.
- VI. A Burmese name with stacked consonants, for which two syllables are kept as one word (**Thinzar**) in Romanization.
- VII. A Burmese name with stacked consonants, for which two syllables are separated as two words (**Thin Zar**) in Romanization. Notice the Burmese names in instance VI. and VII. are identical. They are treated as two different Romanization instances due to the spellings in Romanization are different.

References

1. Banchs, R.E., Zhang, M., Duan, X., Li, H., Kumaran, A.: Report of NEWS 2015 machine transliteration shared task. In: Proceedings of NEWS, pp. 10–23 (2015)
2. Costa-Jussà, M.R.: Moses-based official baseline for NEWS 2016. In: Proceedings of NEWS, pp. 88–90 (2016)
3. Ding, C., Thu, Y.K., Utiyama, M., Finch, A., Sumita, E.: Empirical dependency-based head finalization for statistical Chinese-, English-, and French-to-Myanmar (Burmese) machine translation. In: Proceedings of IWSLT, pp. 184–191 (2014)
4. Ding, C., Thu, Y.K., Utiyama, M., Sumita, E.: Parsing Myanmar (Burmese) by using Japanese as a pivot. In: Proceedings of ICCA (Myanmar), pp. 158–162 (2016)
5. Ding, C., Thu, Y.K., Utiyama, M., Sumita, E.: Word segmentation for Burmese (Myanmar). *ACM Trans. Asian Low Resour. Lang. Inf. Process.* **15**(4), 22 (2016)
6. Finch, A., Liu, L., Wang, X., Sumita, E.: Neural network transduction models in transliteration generation. In: Proceedings of NEWS, pp. 61–66 (2015)
7. Finch, A., Liu, L., Wang, X., Sumita, E.: Target-bidirectional neural models for machine transliteration. In: Proceedings of NEWS, pp. 78–82 (2016)
8. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: Proceedings of ICML, pp. 282–289 (2001)
9. Liu, L., Finch, A., Utiyama, M., Sumita, E.: Agreement on target-bidirectional LSTMs for sequence-to-sequence learning. In: Proceedings of AAAI, pp. 2630–2637 (2016)
10. Naing, H.M.S., Hlaing, A.M., Pa, W.P., Hu, X., Thu, Y.K., Hori, C., Kawai, H.: A Myanmar large vocabulary continuous speech recognition system. In: Proceedings of APSIPA, pp. 320–327 (2015)
11. Neubig, G., Nakata, Y., Mori, S.: Pointwise prediction for robust, adaptable Japanese morphological analysis. In: Proceedings of ACL-HLT, pp. 529–533 (2011)
12. Och, F.J., Ney, H.: A systematic comparison of various statistical alignment models. *Comput. Linguist.* **29**(1), 19–51 (2003)
13. Okell, J.: A guide to the Romanization of Burmese (1971)

14. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: Bleu: a method for automatic evaluation of machine translation. In: Proceedings of ACL, pp. 311–318 (2002)
15. Sha, F., Pereira, F.: Shallow parsing with conditional random fields. In: Proceedings of HLT-NAACT, pp. 134–141 (2003)
16. Thu, Y.K., Pa, W.P., Finch, A., Ni, J., Sumita, E., Hori, C.: The application of phrase based statistical machine translation techniques to Myanmar grapheme to phoneme conversion. In: Hasida, K., Purwarianti, A. (eds.) Computational Linguistics. CCIS, vol. 593, pp. 238–250. Springer, Singapore (2016). https://doi.org/10.1007/978-981-10-0515-2_17
17. Thu, Y.K., Pa, W.P., Ni, J., Shiga, Y., Finch, A., Hori, C., Kawai, H., Sumita, E.: HMM based Myanmar text to speech system. In: Proceedings of INTERSPEECH, pp. 2237–2241 (2015)

Document Classification



Domain Adaptation for Document Classification by Alternately Using Semi-supervised Learning and Feature Weighted Learning

Hiroyuki Shinnou^(✉), Kanako Komiya, and Minoru Sasaki

Ibaraki University, 4-12-1 Nakanarusawa, Hitachi, Ibaraki 316-8511, Japan
{hiroyuki.shinnou.0828,kanako.komiya.nlp,
minoru.sasaki.01}@vc.ibaraki.ac.jp

Abstract. In this paper, we propose a new unsupervised domain adaptation method for document classification. We address the problem of domain adaptation for document classification where the source and target domains do not differ significantly and there is no labeled data in the target domain. In this case, we can use conventional semi-supervised learning. Thus, we use the naive Bayes-based expectation-maximization method (NBEM) which is very effective for document classification. However, NBEM does not utilize the difference between a source domain and a target domain. We combine NBEM with the feature weighted method for domain adaptation, referred to as “self-training feature weight” (STFW). Our proposed method alternately uses NBEM and STFW to gradually improve document classification precision for a target domain. This method significantly outperforms the conventional unsupervised methods for domain adaptation.

Keywords: Domain adaptation · Document classification
Semi-supervised learning · Feature-based methods

1 Introduction

In this paper, we propose a new unsupervised domain adaptation method for document classification. The proposed method alternately uses semi-supervised learning and feature weighted learning to improve document classification precision for a target domain.

Supervised learning has been successfully applied to several natural language processing (NLP) tasks. However, supervised learning can encounter difficulties in real applications when the test data domain (target domain) differs from the training data domain (source domain), i.e. *domain adaptation problem* [13].

A domain adaptation method has two types; supervised and unsupervised. In supervised methods, some labeled data in the target domain is available, but not in unsupervised methods. As a supervised method, Daumé’s method [6]

is very simple and powerful, so this method is essentially standard. However, as an unsupervised method, we do not have one-size-fits-all solution. Thus an unsupervised adaptation method is currently being intensively investigated.

Generally, the domain adaptation problem is handled by instance-based or feature-based methods [10]. This is true in both of supervised cases and unsupervised cases. Instance-based methods assign a weight (importance) to an instance. *Learning under covariate shift* is the most commonly used method [14]. Covariate shift assumes that $P_S(y|\mathbf{x}) = P_T(y|\mathbf{x})$ but $P_S(\mathbf{x}) \neq P_T(\mathbf{x})$. Essentially, learning under covariate shift is loss function based learning with the probability density ratio $P_T(\mathbf{x})/P_S(\mathbf{x})$ as a weight. On the contrary, the feature-based method assigns a weight to a feature. This method maps the feature space in a source domain and a target domain into a common feature space. This mapping is a type of dimension reduction that reduces the difference between the source and target domains but retains the important features of both. Blitzer defined a *pivot feature* as a feature that frequently occurs in both the source and target domains, and proposed *structural correspondence learning (SCL)* for dimension reduction using pivot features [1].

The above methods are effective for some domain adaptation problems. However, they are not as effective in an unsupervised situation, precision reduces. Especially, it is so when the target and source domain are similar like some domain adaptations for document classification. When target and source domains do not differ significantly, we regard the domain adaptation problem as a data sparseness problem. In this case, we can use conventional semi-supervised learning [3] and active learning [12] without modification.

In this paper, we address the problem of domain adaptation for document classification where difference between the source and target domains is small. Therefore, as explained previously, semi-supervised learning is available. A naive Bayes expectation-maximization (NBEM) approach is effective for semi-supervised learning for document classification [9]. Thus, we use NBEM for our domain adaptation problem. However, NBEM does not utilize the difference between a source domain and a target domain. Therefore, ways to improve the process should be considered when NBEM is used for domain adaptation problems. We combine NBEM with an feature weighted method (referred to as “self-training feature weight” (STFW)) which is a feature-based method for domain adaptation. Our proposed method uses NBEM and STFW alternately to gradually improve the precision of document classification for a target domain.

In an experiment, we used data from the 20 Newsgroups dataset to set document classification tasks for domains X and Y . We compared the proposed method to conventional methods for domain adaptation $X \rightarrow Y$ and $Y \rightarrow X$ for document classifications. The results show that the proposed method achieves much higher precision.

2 Related Work

Some studies have used NBEM for domain adaptation of document classification. The NB transfer classifier (NBTC) modifies the EM component of NBEM

to adapt to a target domain [5]. NBTC requires the probability that a test document appears in the source domain. NBTC estimates this probability using the Kullback-Leibler divergence between source and target domains and empirical parameters. Similar to NBTC, the adapting NB (ANB) approach also modifies the EM components in NBEM [15]. ANB uses a mixture distribution of the source and target domains as a document generative model. The weight of the source domain is reduced according to EM iterations. As a result, both NBEM and ANB give weight to a feature through the class distribution of the target domain. On the other hand, the proposed method is based on the idea that the feature must be weighted if the class distribution of a feature in the target domain is similar.

Chen’s method, named as CODA [4], is very similar to our proposed method. The CODA is a kind of co-training [2]. The CODA gives weights to features using the current classifier for target domain. This is a kind of feature-based domain adaptation method. Then using new training data, the classifier is improved by a general learning method. These two procedures are alternately used. The essential difference between our method and CODA is the way to construct the classifier. In that way, we use NBEM, that is, unlabeled data in addition to labeled data are used. On the other hand, CODA does not use unlabeled data. Moreover, used feature-based domain adaptation methods are different. The CODA uses the correlation coefficient between a label and a feature to get feature weights. This means that CODA handles only binary classification. On the other hand, our method uses the label distribution on the feature to get feature weights. Thus, our method has no above limits.

3 Alternating Use of NBEM and STFW

3.1 NBEM

NBEM is a semi-supervised learning method that learns a classifier using a small amount of labeled data and a large amount of unlabeled data. Generally, NBEM learns a classifier using an NB method and labeled data. Then, the classifier is improved by an EM algorithm and unlabeled data.

In this paper, we show only the key equations and the key algorithm for this method [9].

Essentially, the method computes $P(f_i|c_j)$ where f_i is a feature and c_j is a class. This probability is given as follows¹:

$$P(f_i|c_j) = \frac{1 + \sum_{k=1}^{|D|} N(f_i, d_k)P(c_j|d_k)}{|F| + \sum_{m=1}^{|F|} \sum_{k=1}^{|D|} N(f_m, d_k)P(c_j|d_k)}. \quad (1)$$

D : all data consisting of labeled data and unlabeled data

d_k : an element in D

F : the set of all features

¹ This equation is smoothed by considering the frequency 0.

f_m : an element in F

$N(f_i, d_k)$: the number of f_i in instance d_k .

If d_k is labeled, $P(c_j|d_k)$ is 0 or 1. If d_k is unlabeled, $P(c_j|d_k)$ is initially 0, and is updated to an appropriate value step by step in proportion to the iteration of the EM algorithm.

By using Eq. (1), the following classifier is constructed:

$$P(c_j|d_i) = \frac{P(c_j) \prod_{f_n \in K_{d_i}} P(f_n|c_j)}{\sum_{r=1}^{|C|} P(c_r) \prod_{f_n \in K_{d_i}} P(f_n|c_r)}. \quad (2)$$

In this equation, K_{d_i} is the set of features in the instance d_i .

$P(c_j)$ is computed by

$$P(c_j) = \frac{1 + \sum_{k=1}^{|D|} P(c_j|d_k)}{|C| + |D|}. \quad (3)$$

The EM algorithm computes $P(c_j|d_i)$ using Eq. (2) (E-step). Next, using Eq. (1), $P(f_i|c_j)$ is computed (M-step). By iterating E-step and M-step, $P(f_i|c_j)$ and $P(c_j|d_i)$ converge. In our experiment, when the difference between the current $P(f_i|c_j)$ and the updated $P(f_i|c_j)$ becomes less than 8×10^{-6} or the iteration number reaches 10, we consider that the algorithm has converged.

3.2 STFW

In this paper, we combine NBEM and a feature-based method for domain adaptation. As the feature-based method, we improve the method used in the CODA. We name the improved method as STFW.

First, we explain the method used in the CODA. This method assigns a weight (importance) to a feature as follows. Let the value of the feature f in data \mathbf{x} be x_f , and the class label of \mathbf{x} be y_x . If \mathbf{x} is labeled data in a source domain, y_x is given. Thus, we can compute the correlation coefficient $\rho_S(x_f, y_x)$ between x_f in a source domain and y_x . If \mathbf{x} is data in a target domain, y_x is unknown; thus, it is estimated by the current classifier as y'_x (i.e. self-training). As a result, we obtain the correlation coefficient $\rho_T(x_f, y'_x)$ between x_f in the target domain and y'_x . The weight $w(f)$ of the feature f is defined as follows:

$$w(f) = \frac{1 + \rho_S(x_f, y_x) \rho_T(x_f, y'_x)}{2}. \quad (4)$$

x_f is updated as follows:

$$x_f^{(t+1)} = (1 + w(f)^{(t)}) \cdot x_f^{(t)}.$$

The CODA uses the correlation coefficient $\rho_S(x_f, y_x)$ and $\rho_T(x_f, y'_x)$ to define the feature weight. Therefore, if the label is categorical, this method is restricted to binary classification problems. To overcome this issue, we improve the method

used in the CODA. Essentially, the original method measures the similarity between the label distribution P_s for the source domain on the feature f and the label distribution P_t for the target domain on the feature f . Thus, we define the distance $d(f)$ between P_s and P_t as follows:

$$d(f) = |P_s - P_t| \quad (5)$$

We show the way how to calculate the $d(f)$ using the following example. Assume the label set is $\{a, b, c\}$ and f is a feature. We pick up all data x with $x_f > 0$ from the source domain labeled data set. Now we assume such data are $x^{(1)}$, $x^{(2)}$, $x^{(3)}$ and $x^{(4)}$ where $x_f^{(1)} = 1$, $x_f^{(2)} = 4$, $x_f^{(3)} = 3$, and $x_f^{(4)} = 2$. And the label of $x^{(1)}$, $x^{(2)}$, $x^{(3)}$ and $x^{(4)}$ is a , b , c and c respectively. In this case, $P_s(a) = 1/10$, $P_s(b) = 4/10$ and $P_s(c) = 5/10$. By the same way, we pick up all data z with $z_f > 0$ from the target domain. Now we assume such data are $z^{(1)}$ and $z^{(2)}$ where $z_f^{(1)} = 4$, $z_f^{(2)} = 5$. And the estimated label of $z^{(1)}$ and $z^{(2)}$ is b and c respectively. In this case, $P_t(a) = 0$, $P_t(b) = 4/9$ and $P_t(c) = 5/9$. Therefore,

$$d(f) = |1/10 - 0| + |4/10 - 4/9| + |5/10 - 5/9| = 0.1999.$$

Next, using $d(f)$, we set the weight of the feature f . Note that our task is document classification, and we use the NB method as a learning algorithm. Thus, the weight of feature f is generally the frequency, which is a non-negative integer value. From this perspective, we set the weight to a non-negative integer value. As a result, using $d(f)$, x_f is updated as follows:

$$x_f^{(t+1)} = \begin{cases} \min(x_f^{(t)} + 1, 30) & d(f) < \theta_1 \\ \max(x_f^{(t)} - 1, 0) & d(f) > \theta_2 \\ x_f^{(t)} & \text{others} \end{cases} \quad (6)$$

In this paper, we set $\theta_1 = 0.2$ and $\theta_2 = 1.5$ ².

In the above example, $x_f^{(1)}$, $x_f^{(2)}$, $x_f^{(3)}$, and $x_f^{(4)}$ are updated to 2, 5, 4, and 3 respectively.

Note that P_t is unknown because data in the target domain do not have a label. Chen's method selects some target domain data with reliable labels by self-training and estimates P_t using these data and labels. Our method also uses self-training, i.e. a classifier learned by NBEM, however our method uses both reliable data and all unlabeled data in the target domain to estimate P_t .

3.3 Alternate Use of NBEM and STFW

In this paper, we propose a method that alternately uses NBEM and STFW (Fig. 1). The proposed method is conducted as follows:

² We set these values according to the result of preliminary experiment.

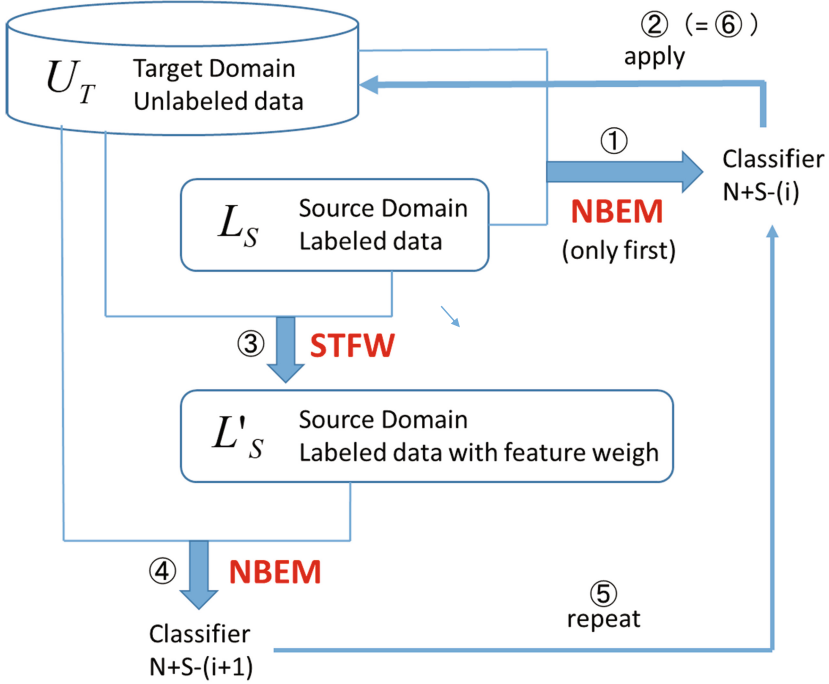


Fig. 1. Alternate use of NBEM and STFW

- (1) The classifier is learned using NBEM with labeled data for a source domain L_S and unlabeled data in a target domain U_T .
- (2) Using the obtained classifier (1), the labels of all data in U_T are estimated.
- (3) Using STFW with the estimated labels (2), new labeled data L'_S is constructed using L_s .
- (4) Return to (1) and set L_S in (1) to L'_s obtained in (3).

We denote the classifier learned through (1) of the i -th repetition as N+S-(i). In this paper, we repeat the above procedure 20 times; thus, we have N+S-(0) to N+S-(20). Note that N+S-(0) is equal to NBEM, N+S-(1) is a simple combined NBEM and STFW method, and N+S-(20) is the final classifier obtained by repeating this combination of NBEM and STFW.

4 Experiments

In the experiment, we picked six categories from the 20 Newsgroups dataset³ as follows. Note that the word in parentheses indicates the class label.

³ <http://qwone.com/~jason/20Newsgroups/>.

A: `comp.sys.ibm.pc.hardware` (comp)
 B: `rec.sport.baseball` (rec)
 C: `sci.electronics` (sci)
 D: `comp.sys.mac.hardware` (comp)
 E: `rec.sport.hockey` (rec)
 F: `sci.med` (sci)

We consider the dataset (A,B,C) as the domain X , and the dataset (D,E,F) as the domain Y . Each domain is the dataset for the document classification task with the class label set $\{\text{comp}, \text{rec}, \text{sci}\}$. In this experiment, we deal with two domain adaptations (i.e. $X \rightarrow Y$ and $Y \rightarrow X$) for document classification.

Table 1 shows the number of documents in each category. In the adaptation of both domains, the label distribution in the training data is uniform. However, the label distribution in the test data is not uniform, and is not equal to each domain adaptation.

Table 1. Number of documents in each set

	Labeled data	Unlabeled data	Test data
A	100	400	300
B	100	300	200
C	100	200	100
D	100	200	100
E	100	400	300
F	100	300	200

In the $X \rightarrow Y$ domain adaptation, the labeled data in the set A, B, and C comprise the training data (300 documents total), and the unlabeled data in the set D, E and F are available for learning (900 documents total). In addition, the test data in the set D, E and F comprise the test data (600 documents total) in the $X \rightarrow Y$ domain adaptation. On the other hand, in the $Y \rightarrow X$ domain adaptation, the labeled data in the set D, E and F comprise the training data (300 documents total), and the unlabeled data in the set A, B and C are available for learning (900 documents total). Moreover, the test data in the set A, B and C comprise the test data (600 documents total) for the $Y \rightarrow X$ domain adaptation.

Table 2 shows the results of the experiment, i.e., precision for the test data obtained using the following methods; NB (S-Only), NBEM, N+S-(20), and NB (T-Only).

Table 2. Experiment result (precision %)

	NB (S-only)	NBEM	N+S-(1)	N+S-(20)	NB (T-only)
X → Y	72.83	90.00	92.33	94.00	94.67
Y → X	81.17	82.67	82.83	87.17	90.00

Note that NB (S-Only) and NB (T-Only) are NB methods that learn using only labeled data in the source domain and only labeled data in the target domain, respectively. In addition, the precision of NB (T-Only) is shown for reference. If the labeled data in the target domain are available, the task is not only domain adaptation but also a general classification problem. Here, N+S-(20) indicates the proposed method. Table 2 shows that the proposed method is very effective.

Figures 2 and 3 show the precision curve for the reputation of NBEM and STFW, where the horizontal axis is the i of N+S-(i). Our method stops after 20 repetitions to obtain the maximum precision.

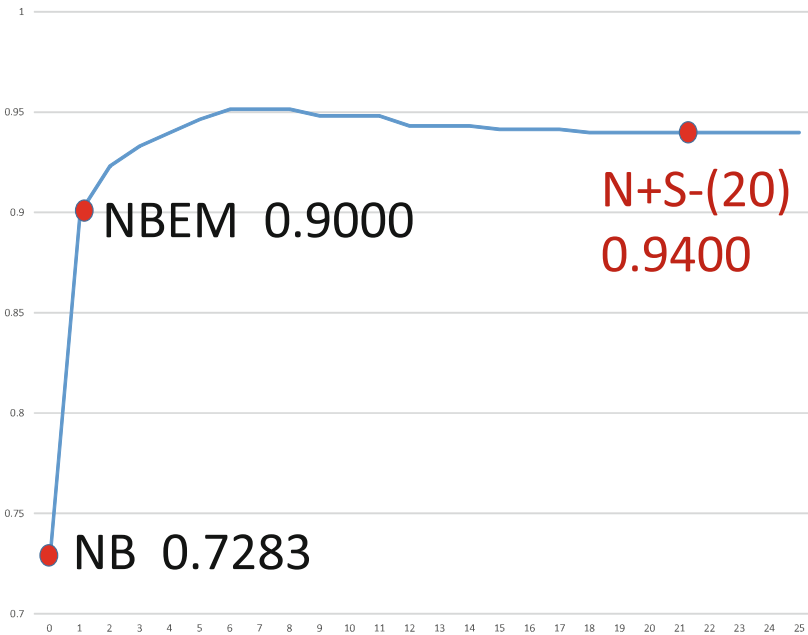


Fig. 2. Precision curve of N+S-(i) (X → Y)

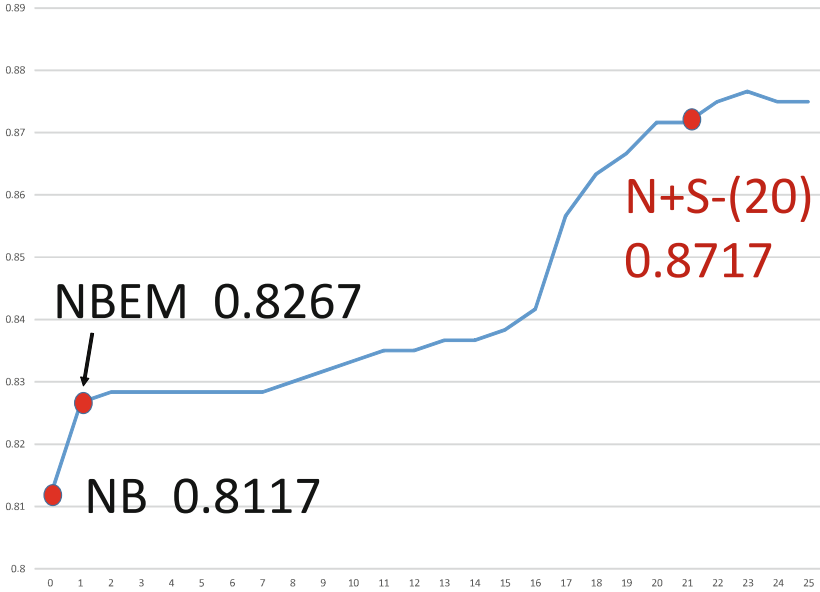


Fig. 3. Precision curve of N+S-(i) ($Y \rightarrow X$)

5 Discussion

5.1 Comparison to Transductive Method

Like semi-supervised learning, transductive learning is another method that uses unlabeled data to improve a classifier learned from labeled data. Transductive-SVM (TSVM) [7] is a representative transductive learning method.

In this paper, although we employ NBEM for semi-supervised learning, it is possible to use TSVM rather than NBEM. Here, we compare NBEM with TSVM. Table 3 shows the results.

Table 3. Comparison to methods using unlabeled data

	NB	NBEM	SVM	TSVM
$X \rightarrow Y$	72.83	90.00	75.83	66.50
$Y \rightarrow X$	81.17	82.67	71.16	70.83

Generally, a SVM has higher precision than the NB. However, NB sometimes demonstrates high precision for document classification. In fact, for the $Y \rightarrow X$ domain adaption, NB outperforms the SVM. When using NB for document classification, it is preferable that documents are simply represented by a bag-of-words. Thus, when using a SVM, it is necessary to perform some pre-processing.

In the previously described experiment with the SVM, we set the vector value by TF * IDF, and normalize the size of the vector to 1.

Note that TSVM reduces the precision of the SVM because TSVM assumes that the class distributions of test and training data are the same, however, this assumption was not satisfied in our experiments.

5.2 Comparing Other Domain Adaption Methods

Domain adaption methods can be classified as feature-based methods and the instance-based methods. Here, we compare a feature-based method and an instance-based method to the proposed method.

We used the representative SCL feature-based method [1] and learning under covariate shift as a typical instance-based method. With learning under covariate shift, the calculation of the probability density ratio becomes the key point. Here, we use the Unconstrained Least Squares Importance Fitting (uLSIF) density calculation method [8].

The experimental results are shown in Table 4. N+S-(1) is the combined NBEM and STFW method. Note that N+S-(1) is initial of our proposed method N+S-(20), and N+S-(20) is much better than N+S-(1).

Table 4. Domain adaptation methods

	N+S-(1)	SVM	SCL	uLSIF
X → Y	92.33	75.83	74.33	73.67
Y → X	82.83	71.16	71.83	72.17

Note that SCL and uLSIF do not significantly improve the precision of the SVM, which is the base system of both methods. On the other hand, N+S-(1) significantly outperforms the SVM because the base system of both SCL and uLSIF is an SVM, and the base system of N+S-(1) is NB. As mentioned previously, NB outperforms the SVM for our task. Moreover, both SCL and uLSIF are transductive methods that use test data in a target domain for learning, however, they do not use unlabeled data. On the other hand, N+S-(1) uses not test data but unlabeled data. Test data is also unlabeled data, but the former is smaller than the latter. In this experiment, the amount of unlabeled data was 1.5 times the amount of the test data. Therefore this can be considered one reason that N+S-(1) is better than SCL and uLSIF.

5.3 Feature Weighting

In this paper, we add weight to a feature that is likely to be valid for classification in domain adaption, and reduce the weight of a feature that is likely to have an adverse effect on classification. Here, we examine the following points.

- Weighting test data
- Size of the added weight
- Negative weights

Our experiment results are discussed in the following.

We set the weight for features of the training data only; however, it is also conceivable to set weights for the test data. The experimental results are shown in Table 5.

Table 5. Weighting test data (TW)

	N+S-(20) (without TW) -proposed method-	N+S-(20) (with TW)
X \rightarrow Y	94.00	89.50
Y \rightarrow X	87.17	85.67

In this paper, giving weight means to plus 1, here we change it to plus 2, and experimental results are shown in Table 6.

Table 6. Changing the size of the added weight

	N+S-(20) (+1) -proposed method-	N+S-(20) (+2)
X \rightarrow Y	94.00	94.00
Y \rightarrow X	87.17	88.17

In domain adaption, there may be some labeled data that create adverse learning results. This is called “negative transfer” [11]. The proposed method is designed based on negative transfer, i.e. if the difference between the class distributions of features in the source and target domains is quite large, we assign a negative feature weight (-1). To investigate the effects of negative weights, we conducted an experiment in which negative weights were not assigned. The experimental results are shown in Table 7.

Table 7. Effects of negative weight (NW)

	N+S-(20) (with NW) -proposed method-	N+S-(20) (without NW)
X \rightarrow Y	94.00	93.33
Y \rightarrow X	87.17	88.67

Through the above experiments, we can confirm that weighting test data is ineffective, and that other changes relative to setting weights have little influence on precision. In future, we will investigate a better method to determine setting weights.

6 Conclusion

In this paper, we have proposed a new method for unsupervised domain adaptation for document classification. The proposed method alternately uses NBEM, i.e. a semi-supervised learning method, and STFW, i.e. a feature based method, for domain adaptation. In an experiment using part of the 20 Newsgroups dataset, the proposed method demonstrated high effectiveness. Note that the proposed method has some meta parameters, and in future, we will investigate a way to set such meta parameters appropriately.

Acknowledgment. The work reported in this article was supported by the NINJAL collaborative research project ‘Development of all-words WSD systems and construction of a correspondence table between WLSP and IJD by these systems.’

References

1. Blitzer, J., McDonald, R., Pereira, F.: Domain adaptation with structural correspondence learning. In: EMNLP-2006, pp. 120–128 (2006)
2. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: Proceedings of the Eleventh Annual Conference on Computational Learning Theory, pp. 92–100 (1998)
3. Chapelle, O., Schölkopf, B., Zien, A., et al.: *Semi-supervised Learning*, vol. 2. MIT Press, Cambridge (2006)
4. Chen, M., Weinberger, K.Q., Blitzer, J.: Co-training for domain adaptation. In: NIPS, pp. 2456–2464 (2011)
5. Dai, W., Xue, G.R., Yang, Q., Yu, Y.: Transferring naive Bayes classifiers for text classification. In: AAI-2007 (2007)
6. Daumé III, H.: Frustratingly easy domain adaptation. In: ACL-2007, pp. 256–263 (2007)
7. Joachims, T.: Transductive inference for text classification using support vector machines. In: ICML, vol. 99, pp. 200–209 (1999)
8. Kanamori, T., Hido, S., Sugiyama, M.: A least-squares approach to direct importance estimation. *J. Mach. Learn. Res.* **10**, 1391–1445 (2009)
9. Nigam, K., McCallum, A., Thrun, S., Mitchell, T.: Text classification from labeled and unlabeled documents using EM. *Mach. Learn.* **39**(2/3), 103–134 (2000)
10. Pan, S.J., Yang, Q.: A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.* **22**(10), 1345–1359 (2010)
11. Rosenstein, M.T., Marx, Z., Kaelbling, L.P., Dietterich, T.G.: To transfer or not to transfer. In: NIPS 2005 Workshop on Transfer Learning, vol. 898 (2005)
12. Settles, B.: *Active Learning Literature Survey*. University of Wisconsin, Madison (2010)
13. Søgaard, A.: *Semi-supervised Learning and Domain Adaptation in Natural Language Processing*. Morgan & Claypool, San Rafael (2013)
14. Sugiyama, M., Kawanabe, M.: *Machine Learning in Non-stationary Environments: Introduction to Covariate Shift Adaptation*. MIT Press, Cambridge (2011)
15. Tan, S., Cheng, X., Wang, Y., Xu, H.: Adapting naive Bayes to domain adaptation for sentiment analysis. In: Boughanem, M., Berrut, C., Mothe, J., Soule-Dupuy, C. (eds.) ECIR 2009. LNCS, vol. 5478, pp. 337–349. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-00958-7_31

Information Extraction and Text Mining



End-to-End Recurrent Neural Network Models for Vietnamese Named Entity Recognition: Word-Level Vs. Character-Level

Thai-Hoang Pham¹(✉) and Phuong Le-Hong²

¹ R&D Department, Alt Inc., Hanoi, Vietnam
phamthaihoang.hn@gmail.com

² College of Science, Vietnam National University, Hanoi, Vietnam
phuonglh@vnu.edu.vn

Abstract. This paper demonstrates end-to-end neural network architectures for Vietnamese named entity recognition. Our best model is a combination of bidirectional Long Short-Term Memory (Bi-LSTM), Convolutional Neural Network (CNN), Conditional Random Field (CRF), using pre-trained word embeddings as input, which achieves an F_1 score of 88.59% on a standard test set. Our system is able to achieve a comparable performance to the first-rank system of the VLSP campaign without using any syntactic or hand-crafted features. We also give an extensive empirical study on using common deep learning models for Vietnamese NER, at both word and character level.

Keywords: Vietnamese · Named entity recognition · End-to-end Long Short-Term Memory · Conditional Random Field Convolutional Neural Network

1 Introduction

Named entity recognition (NER) is a fundamental task in natural language processing and information extraction. It involves identifying noun phrases and classifying each of them into a predefined class. In 1995, the 6th Message Understanding Conference (MUC)¹ started evaluating NER systems for English, and in subsequent shared tasks of CoNLL 2002² and CoNLL 2003³ conferences, language independent NER systems were evaluated. In these evaluation tasks, four named entity types were considered, including names of persons, organizations, locations, and names of miscellaneous entities that do not belong to these three types.

¹ <http://cs.nyu.edu/faculty/grishman/muc6.html>.

² <http://www.cnts.ua.ac.be/conll2002/ner/>.

³ <http://www.cnts.ua.ac.be/conll2003/ner/>.

More recently, the Vietnamese Language and Speech Processing (VLSP)⁴ community has organized an evaluation campaign to systematically compare NER systems for the Vietnamese language. Similar to the CoNLL 2003 share task, four named entity types are evaluated: persons (PER), organizations (ORG), locations (LOC), and miscellaneous entities (MISC). The data are collected from electronic newspapers published on the web.

In this paper, we present a state-of-the-art NER system for the Vietnamese language without using any hand-crafted features. Our system is competitive with the first-rank system of the VLSP campaign that used many syntactic and hand-crafted features. In summary, the overall F_1 score of our system is 88.59% on the standard test set provided by the organizing committee of the evaluation campaign⁵. The contributions of this work include:

- We propose a truly end-to-end deep learning model which gives the state-of-the-art performance on a standard NER data set for Vietnamese. Our best model is a combination of Bi-LSTM, CNN, and CRF models, which achieves an F_1 score of 88.59%.
- We give an extensive empirical study on using common deep learning models for Vietnamese NER, at both word and character level. These models are also comparable to conventional sequence labeling models, including Maximum Entropy Markov Models (MEMMs) and CRFs.
- We make our NER system open source for research purpose, which is believed to be a good contribution to the future development of Vietnamese NER in particular and Vietnamese language processing research in general.

The remainder of this paper is structured as follows. Section 2 summarizes related work on NER. Section 3 describes end-to-end models used in our system. Section 4 gives experimental results and discussions. Finally, Sect. 5 concludes the paper.

2 Related Work

Within the large body of research on NER which have been published in the last two decades, we identify two main approaches. The first approach is characterized by the use of well-established sequence labeling models such as conditional random field (CRF), hidden markov model, support vector machine, maximum entropy and so on. The performance of these models is heavily dependent on hand-crafted features. In particular, most of the participants at CoNLL-2003 shared task attempted to use information other than the available training data such as gazetteers and unannotated data. The best system at CoNLL-2003 shared task is the work of [5] which achieved an F_1 score of 88.76%. After that, [17] surpassed them by using phrase features extracted from an external database. Moreover, training NER models jointly with related tasks helps

⁴ <http://vlsp.org.vn/>.

⁵ The first-rank system of the VLSP 2016 NER evaluation campaign has $F_1 = 88.78\%$ on the test set.

improve their performance. For instance, [4] trained a CRF model for joint-learning three tasks, including coreference resolution, entity linking, and NER, and achieved the state-of-the-art result on OntoNotes dataset. With a similar approach, [18] gained the best performance on CoNLL-2003 shared task dataset.

With a recent resurgence of the deep learning approach, several neural architectures have been proposed for NER task. These methods have a long story, but they have been focused only recently by the advance of computational power and high-quality word embeddings. The first neural network model is the work of [23] that used a feed-forward neural network with one hidden layer. This model achieved the state-of-the-art result on the MUC6 dataset. After that, [8] used a long short-term memory network for this problem. Recently, [3] used a convolution neural network over a sequence of word embeddings with a conditional random field on the top. This model achieved near state-of-the-art results on some sequence labeling tasks such as POS tagging, chunking, and NER. From 2015 until now, the long short-term memory model has been the best approach for many sequence labeling tasks. [10] used bidirectional LSTM with CRF layer for joint decoding. Instead of using hand-crafted feature as [2, 10] proposed a hybrid model that combined bidirectional LSTM with convolutional neural networks (CNN) to learn both character-level and word-level representations. Unlike [2, 13] used bidirectional LSTM to model both character and word-level information. The work of [19] proposed a truly end-to-end model that used only word embeddings for detecting entities. This model is the combination of CNN, bidirectional LSTM, and CRF models. Approaching this problem at the character-level sequence, the LSTM-CRF model of [11] achieved the nearly state-of-the-art results in seven languages.

3 Methodology

3.1 Long Short-Term Memory Networks

Recurrent Neural Network. The recurrent neural network (RNN) is a class of artificial neural network designed for sequence labeling task. It takes input as a sequence of vector and returns another sequence. The simple architecture of RNN has an input layer \mathbf{x} , hidden layer \mathbf{h} and output layer \mathbf{y} . At each time step t , the values of each layer are computed as follows:

$$\begin{aligned}\mathbf{h}_t &= f(\mathbf{U}\mathbf{x}_t + \mathbf{W}\mathbf{h}_{t-1}) \\ \mathbf{y}_t &= g(\mathbf{V}\mathbf{h}_t)\end{aligned}$$

where \mathbf{U} , \mathbf{W} , and \mathbf{V} are the connection weight matrices in RNN, and $f(z)$ and $g(z)$ are sigmoid and softmax activation functions.

Long Short-Term Memory. Long short-term memory (LSTM) [9] is a variant of RNN which is designed to deal with these gradient vanishing and exploding problems [1, 22] when learning with long-range sequences. LSTM networks are

the same as RNN, except that the hidden layer updates are replaced by memory cells. Basically, a memory cell unit is composed of three multiplicative gates that control the proportions of information to forget and to pass on to the next time step. As a result, it is better for exploiting long-range dependency data. The memory cell is computed as follows:

$$\begin{aligned} \mathbf{i}_t &= \sigma(\mathbf{W}_i \mathbf{h}_{t-1} + \mathbf{U}_i \mathbf{x}_t + \mathbf{b}_i) \\ \mathbf{f}_t &= \sigma(\mathbf{W}_f \mathbf{h}_{t-1} + \mathbf{U}_f \mathbf{x}_t + \mathbf{b}_f) \\ \mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tanh(\mathbf{W}_c \mathbf{h}_{t-1} + \mathbf{U}_c \mathbf{x}_t + \mathbf{b}_c) \\ \mathbf{o}_t &= \sigma(\mathbf{W}_o \mathbf{h}_{t-1} + \mathbf{U}_o \mathbf{x}_t + \mathbf{b}_o) \\ \mathbf{h}_t &= \mathbf{o}_t \odot \tanh(\mathbf{c}_t) \end{aligned}$$

where σ is the element-wise sigmoid function and \odot is the element-wise product, \mathbf{i} , \mathbf{f} , \mathbf{o} and \mathbf{c} are the input gate, forget gate, output gate and cell vector respectively. $\mathbf{U}_i, \mathbf{U}_f, \mathbf{U}_c, \mathbf{U}_o$ are connection weight matrices between input \mathbf{x} and gates, and $\mathbf{W}_i, \mathbf{W}_f, \mathbf{W}_c, \mathbf{W}_o$ are connection weight matrices between gates and hidden state \mathbf{h} . $\mathbf{b}_i, \mathbf{b}_f, \mathbf{b}_c, \mathbf{b}_o$ are the bias vectors.

Bidirectional Long Short-Term Memory. The original LSTM uses only previous contexts for prediction. For many sequence labeling tasks, it is advisable when taking the contexts from two directions. Thus, we utilize the bidirectional LSTM (Bi-LSTM) [6, 7] for both word and character-level systems.

3.2 Conditional Random Field

Conditional Random Field (CRF) [12] is a type of graphical model designed for labeling sequence of data. Although the LSTM is likely to handle the sequence of the input data by learning the dependencies between the input at each time step but it predicts the outputs independently. The CRF, therefore, is beneficial to explore the correlations between outputs and jointly decode the best sequence of labels. In NER task, we implement the CRF on the top of Bi-LSTM instead of the softmax layer and take outputs of Bi-LSTM as the inputs of this model. The parameter of the CRF is the transition matrix A where $A_{i,j}$ represents the transition score from tag i to tag j . The score of the input sentence \mathbf{x} along with the sequence of tags \mathbf{y} is computed as follow:

$$S(\mathbf{x}, \mathbf{y}, \theta \cup A_{i,j}) = \sum_{t=1}^T (A_{y_{t-1}, y_t} + f_{\theta(y_t, t)})$$

where θ is the parameters of Bi-LSTM, f_{θ} is the score outputted by Bi-LSTM, and T is the number of time steps. Then the tag-sequence likelihood is computed by the softmax equation:

$$p(\mathbf{y}|\mathbf{x}, A) = \frac{\exp(S(\mathbf{x}, \mathbf{y}, \theta \cup A_{i,j}))}{\sum_{\mathbf{y}' \in \mathbf{Y}} \exp(S(\mathbf{x}, \mathbf{y}', \theta \cup A_{i,j}))}$$

where \mathbf{Y} is the set of all possible output sequences. In the training stage, we maximize the log-likelihood function:

$$L = \sum_{i=1}^N \log p(\mathbf{y}^i | \mathbf{x}^i; A)$$

where N is the number of training samples. In the inference stage, the Viterbi algorithm is used to find the output sequence \mathbf{y}^* that maximize the conditional probability:

$$\mathbf{y}^* = \arg \max_{\mathbf{y} \in \mathbf{Y}} p(\mathbf{y} | \mathbf{x}, A)$$

3.3 Learning Word Embeddings

It has been shown that distributed representations of words (words embeddings) help improve the accuracy of a various natural language models. In this work, we investigate three methods to create word embeddings using a skip-gram model, a CNN model and a Bi-LSTM model.

Pre-Trained Word Vectors Learnt by Skip-Gram Model. To create word embeddings for Vietnamese, we train a skip-gram model using the word2vec⁶ tool on a dataset consisting of 7.3 GB of text from 2 million articles collected through a Vietnamese news portal.⁷ The text is first normalized to lower case and all special characters are removed. The common symbols such as the comma, the semicolon, the colon, the full stop and the percentage sign are replaced with the special token *punct*, and all numeral sequences are replaced with the special token *number*. Each word in the Vietnamese language may consist of more than one syllables with spaces in between, which could be regarded as multiple words by the unsupervised models. Hence it is necessary to replace the spaces within each word with underscores to create full word tokens. The tokenization process follows the method described in [16]. For words that appear in VLSP corpus but not appear in word embeddings set, we create random vectors for these words by uniformly sampling from the range $[-\sqrt{\frac{3}{dim}}, +\sqrt{\frac{3}{dim}}]$ where *dim* is the dimension of embeddings.

Character-Level Word Vectors Learnt by Convolutional Neural Network. Convolutional neural network (CNN) is a type of feed-forward neural networks that uses many identical copies of the same neuron. This characteristic of CNN permits this network to have lots of neurons and, therefore, express computationally large models while keeping the number of actual parameters relatively small. For NLP tasks, previous works have shown that CNN is likely to extract morphological features such as prefix and suffix effectively [2, 19, 24].

⁶ <https://code.google.com/archive/p/word2vec/>.

⁷ <http://www.baomoi.com>.

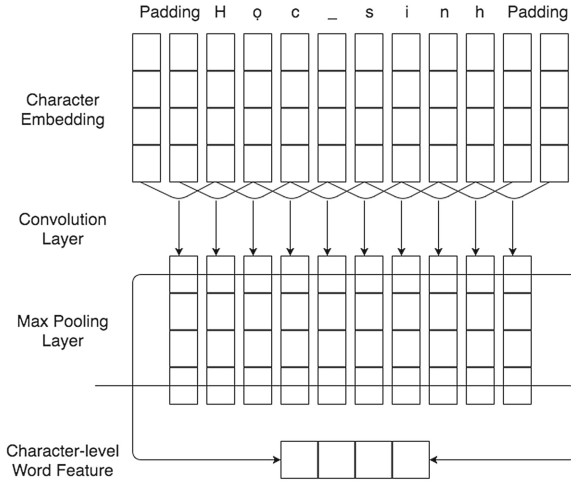


Fig. 1. The CNN for extracting character-level word features of word *Hoc_sinh* (Student).

For this reason, we incorporate the CNN to the word-level model to get richer information from character-level word vectors. These vectors are learnt during training together with the parameters of the word models. The CNN we use in this paper is described in Fig. 1.

Character-Level Word Vectors Learnt by Long Short-Term Memory.

The second way for generating character-level word vectors is using Bi-LSTM. In particular, we incorporate this model to the word-level model to learn character-level word vectors. Character-level word vectors are concatenations of two last hidden states from forward and backward layers of Bi-LSTM. These vectors are also learnt during training together with the parameters of the word models. The Bi-LSTM model we use for this task is described in Fig. 2.

3.4 Our Proposed Models

We propose two different types of models based on the level of input, either using word sequence or character sequence. Concretely, in the first type, each input sentence is fed to the model as a sequence of words, while in the second type, it is fed as a sequence of characters. Both of the two model types share the same pipeline in that it takes as input a sequence of distributed representations of the underlying processing unit (word or character), that sequence is then passed to a Bi-LSTM, and then a CRF layer takes as input the output of the Bi-LSTM to predict the best named entity output sequence.

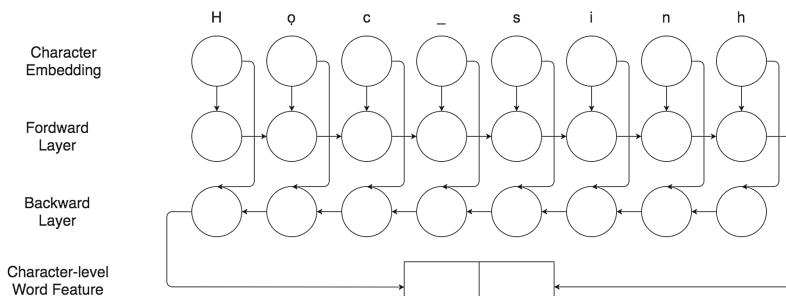


Fig. 2. The Bi-LSTM for extracting character-level word features of word *Học sinh* (Student).

Word-Levels Models. In the first type, we investigate four different word embeddings, including (**Word-0**) random vectors, (**Word-1**) skip-gram vectors, (**Word-2**) skip-gram vectors concatenated with CNN-generated word features, and (**Word-3**) skip-gram vectors concatenated with LSTM-generated word features. Figure 3 describes the architecture of the word-level models.

Character-Level Model. In the second type, we investigate one model in that its input is a sequence of vectors corresponding to characters of the input sentence. We call this model (**Char-0**). Because the size of Vietnamese character set is relatively small, our data set is sufficient to learn distributed representations for Vietnamese characters. We therefore initialize random vectors for these characters by uniformly sampling from the range $[-\sqrt{\frac{3}{dim}}, +\sqrt{\frac{3}{dim}}]$ where dim is the dimension of embeddings. These character vectors are then learnt during training together with the parameters of the models.

The training data for NER is in CoNLL-2003 format, where both input and output sequence are annotated at word-level. For this reason, it is necessary to convert the dataset from word-level sequences to character-level sequences. We use a simple method in which all characters of a word are labeled with the same tag. For example, the label of all characters of a person named entity is P. Similarly, all characters of location, organization, and miscellaneous tokens are labelled with letters L, G, and M respectively. The characters of other words and spaces are labelled by O. Figure 4 shows the transformation from word-level to character-level of an example sentence *Anh rời EU hôm qua* (UK left EU yesterday) and Fig. 5 describes the architecture of the character-level models.

4 Results and Discussions

4.1 VLSP Corpus

We evaluate our system on the VLSP NER shared task 2016 corpus. This corpus consists of electronic newspapers published on the web. There are four named

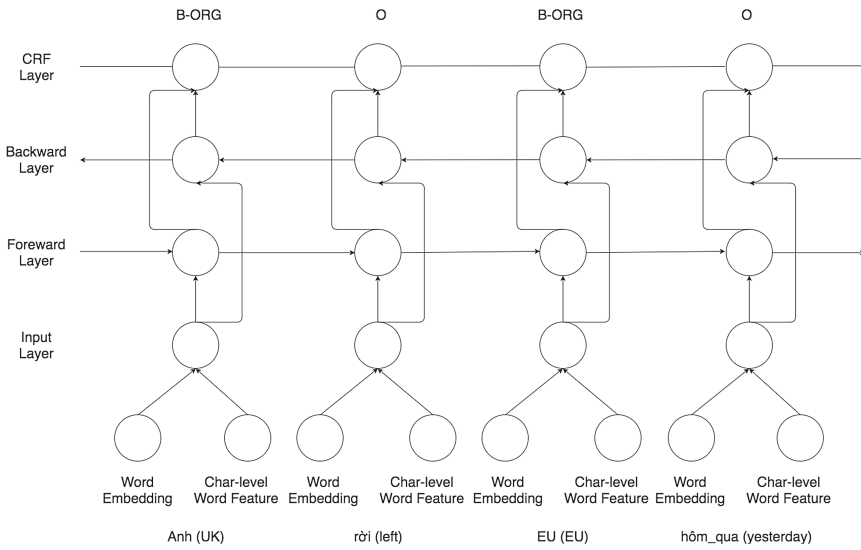


Fig. 3. Word-level model type for input sentence *Anh rời EU hôm qua* (UK left EU yesterday.) **Word-0** and **Word-1** models uses only word embeddings as input, while **Word-2** and **Word-3** models uses both word embeddings and word features generated either by CNN or Bi-LSTM.

Anh rời EU hôm_qua
 B-ORG O B-ORG O

 A n h r ờ i E U h ô m _ q u a
 G G G O O O O O G G O O O O O O O O

Fig. 4. Word and character-level sequence labeling of the sentence *Anh rời EU hôm_qua*. (UK left EU yesterday.)

entity types in this corpus, names of person, location, organization and other named entities. Four types of NEs are compatible with their descriptions in the CoNLL shared task 2003. The examples of each entity type are described in Table 1.

Data have been preprocessed with word segmentation and POS tagging. Because POS tags and chunking tags are determined automatically by public tools, they may contain mistakes. The format of this corpus follows that of the CoNLL 2003 shared task. It consists of five columns. The order of these columns are word, POS tag, chunking tag, named entity label, and nested named entity label. Our system focuses on only named entity without nesting, so we do not use the fifth column. Named entity labels are annotated using the IOB notation as in the CoNLL shared tasks. There are 9 labels: B-PER and I-PER are used for persons, B-ORG and I-ORG are used for organizations, B-LOC and I-LOC are used for locations, B-MISC and I-MISC are used for other named entities

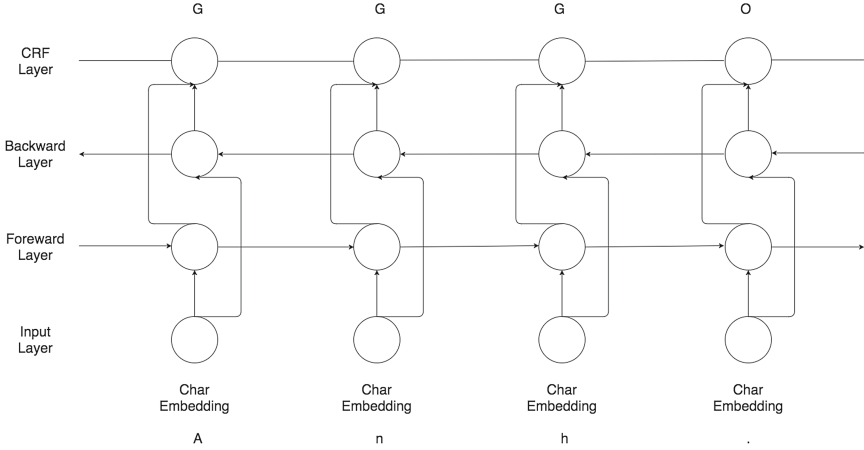


Fig. 5. Character-level model type for input sentence *Anh*. (UK.)

Table 1. Examples of Vietnamese entity types

Entity Types	Examples
Person	thành phố Hồ Chí Minh (Ho Chi Minh city), núi Bà Đen (Ba Den mountain), sông Bạch Đằng (Bach Dang river)
Location	công ty Formosa (Formosa company), nhà máy thủy điện Hòa Bình (Hoa Binh hydroelectric factory)
Organization	ông Lan (Mr. Lan), bà Hà (Mrs. Ha)
Miscellaneous names	tiếng Indonesia (Indonesian), người Canada (Canadian)

and O is used for other elements. Table 2 shows the quantity of named entity annotated in the training set and the test set.

Because our systems are end-to-end architecture, we focus only on the word and named entity label columns. To alleviate the data sparseness, we perform the following preprocessing for our system:

- All tokens containing digit number are replaced by a special token *number*.
- All punctuations are replaced by a special token *punct*.

Moreover, we take one part of training data for validation. The detail of each data set is described in Table 3.

4.2 Evaluation Method

The performance is measured with F_1 score, where $F_1 = \frac{2*P*R}{P+R}$. Precision (P) is the percentage of named entities found by the learning system that are correct.

Table 2. Statistics of named entities in VLSP corpus

Entity types	Training set	Testing set
Location	6,247	1,379
Organization	1,213	274
Person	7,480	1,294
Miscellaneous names	282	49
All	15,222	2,996

Table 3. Size of each data set in VLSP corpus

Data sets	Number of sentences
Train	14,861
Dev	2,000
Test	2,831

Recall (R) is the percentage of named entities present in the corpus that are found by the system. A named entity is correct only if it is an exact match of the corresponding entity in the data file. For character-level model, after predicting label for each character, we convert these outputs back to the word-level sequence to evaluate. The performance of our system is evaluated by the automatic evaluation script of the CoNLL 2003 shared task.⁸

4.3 Results

Word-Level Model Vs. Character-Level Model. In the first experiment, we compare the effectiveness of word and character-level approaches without using any external corpus. For this reason, in this experiment, we do not use any pre-trained word embeddings by comparing two models: **Word-0** and **Char-0**. Both of the two models take embeddings as inputs of Bi-LSTM and predict outputs by the CRF top layer. Table 4 presents the performance of these systems.

Table 4. Performances of word and character-level models

Entity	Word-0			Char-0		
	P	R	F_1	P	R	F_1
LOC	88.37	74.69	80.95	80.03	84.84	82.37
MISC	90.48	77.55	83.52	84.21	65.31	73.56
ORG	60.57	38.83	47.32	50.00	33.58	40.17
PER	89.49	66.51	76.31	84.20	86.09	85.14
ALL	86.78	67.90	76.19	80.08	80.37	80.23

⁸ <http://www.cnts.ua.ac.be/conll2003/ner/>.

We see that the character-level model outperforms the word-level model by about 4%. It is because the size of the character set is much smaller than that of word set. The VLSP corpus, therefore, is enough for learning effectively character embeddings. For word embeddings, we need a bigger corpus to learn useful word vectors.

Effect of Word Embeddings. It is beneficial to use the external corpus to learn the word embeddings. In the second experiment, we use skip-gram word embeddings and compare **Word-1** and **Word-0** models. The improvement by using pre-trained word embeddings for the word-level model is shown in Table 5.

By using pre-trained word embeddings, the performance of word-level model increases by about 4%, to 80.85%. This accuracy is comparable to that of the character-level model. It proves the effectiveness of using good embeddings for both words and characters in the Bi-LSTM-CRF model.

Table 5. Performances of random and word2vec embeddings for word-level model

Entity	Word-0			Word-1		
	<i>P</i>	<i>R</i>	<i>F</i> ₁	<i>P</i>	<i>R</i>	<i>F</i> ₁
LOC	88.37	74.69	80.95	87.88	84.08	85.94
MISC	90.48	77.55	83.52	90.00	73.47	80.90
ORG	60.57	38.83	47.32	72.77	50.92	59.91
PER	89.49	66.51	76.31	88.92	71.38	79.19
ALL	86.78	67.90	76.19	87.21	75.35	80.85

Effect of Character-Level Word Features. In the third experiment, we evaluate the performance of **Word-2** and **Word-3** models. Recall that these two models make use of both pre-trained skip-gram word embeddings and character-level word features generated either by CNN or Bi-LSTM. The obtained performances are described in Table 6.

Table 6. Performances of word-level models

Entity	Word-3			Word-2			Word-1		
	<i>P</i>	<i>R</i>	<i>F</i> ₁	<i>P</i>	<i>R</i>	<i>F</i> ₁	<i>P</i>	<i>R</i>	<i>F</i> ₁
LOC	90.72	88.26	89.48	91.60	88.85	90.20	87.88	84.08	85.94
MISC	94.29	67.35	78.57	97.30	73.47	83.72	90.00	73.47	80.90
ORG	69.23	52.75	59.88	72.77	62.64	67.32	72.77	50.92	59.91
PER	90.12	72.62	80.43	93.60	88.24	90.84	88.92	71.38	79.19
ALL	88.82	77.87	82.98	90.97	85.93	88.38	87.21	75.35	80.85

We observe a significant improvement of performance when character-level word features learnt by CNN are integrated with pre-trained word embeddings. This model achieves an overall F_1 score of 88.38%. The character-level word features learnt by Bi-LSTM are not as good as those learnt by CNN, achieves only an overall F_1 score of 82.98%, but they also help improve the performance of the model in comparison to the **Word-1** model.

Comparison with Previous Systems. In VLSP 2016 workshop, several different systems have been proposed for Vietnamese NER. In this campaign, they have evaluated over three entities types *LOC*, *ORG*, *PER*. In all fairness, we also evaluate our performances over these tags on the same training and test set. The accuracy of our best model over three entity types is 88.59%, which is competitive with the best participating system [15] in that shared task. That system, however, used many hand-crafted features to improve the performance of maximum entropy classifier (ME) while our system is truly end-to-end model that takes only word sequences as inputs. Most approaches in VLSP 2016 used the CRF and ME models, whose performance is heavily dependent on feature engineering. Table 7 shows those models and their performance.

Table 7. Comparison to participating NER systems at VLSP 2016

Team	Model	Performance
[15]	ME	88.78
Word-2	Bi-LSTM-CNN-CRF	88.59
[Anonymous] ^a	CRF	86.62
[20]	ME	84.08
[21]	Bi-LSTM-CRF	83.80
[14]	CRF	78.40

^aThis team provided a system without the technical report.

There is one work [21] that applied deep learning approach for this task. They used the implementation provided by [13]. There are two types of LSTM models in this open source software: Bi-LSTM-CRF and Stack-LSTM. The model that is most similar to ours is Bi-LSTM-CRF. The accuracy of this system is 83.25%. Our system outperforms this model due to some possible reasons. First, they used random vectors as word embeddings and update them during the training stage. The VLSP corpus size is relatively small so it is not good enough for learning word representations. Our word embeddings are trained on a collection of Vietnamese newspapers that is much larger and more abundant than the VLSP corpus. Second, they used LSTM to model character-level features, while we used CNN in our model. Previous works have shown that CNN is very useful to extract these features [2, 19, 24].

5 Conclusion

In this work, we have investigated a variety of end-to-end recurrent neural network architectures at both word and character-level for Vietnamese named entity recognition. Our best end-to-end system is the combination of Bi-LSTM, CNN, and CRF models, and uses pre-trained word embeddings as input, which achieves an F_1 score of 88.59% on the standard test corpus published recently by the Vietnamese Language and Speech community. Our system is competitive with the first-rank system of the related NER shared task without using any hand-crafted features.

Acknowledgement. The second author is partly funded by the Vietnam National University, Hanoi (VNU) under project number QG.15.04. Any opinions, findings and conclusion expressed in this paper are those of the authors and do not necessarily reflect the view of VNU.

References

1. Bengio, Y., Simard, P., Frasconi, P.: Learning long-term dependencies with gradient descent is difficult. *IEEE Trans. Neural Netw.* **5**(2), 157–166 (1994)
2. Chiu, J.P., Nichols, E.: Named entity recognition with bidirectional LSTM-CNNs. *Trans. Assoc. Comput. Linguist.* **4**, 357–370 (2016)
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. Durrett, G., Klein, D.: A joint model for entity analysis: coreference, typing, and linking. *Trans. Assoc. Comput. Linguist.* **2**, 477–490 (2014)
5. Florian, R., Ittycheriah, A., Jing, H., Zhang, T.: Named entity recognition through classifier combination. In: Daelemans, W., Osborne, M. (eds.) *Proceedings of CoNLL-2003*, pp. 168–171. Edmonton, Canada (2003)
6. Graves, A., Mohamed, A.R., Hinton, G.: Speech recognition with deep recurrent neural networks. In: *Proceedings of 2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 6645–6649. IEEE, Vancouver (2013)
7. Graves, A., Schmidhuber, J.: Framewise phoneme classification with bidirectional LSTM networks. In: *Proceedings of 2005 IEEE International Joint Conference on Neural Networks*, vol. 4, pp. 2047–2052. IEEE, Montreal (2005)
8. Hammerton, J.: Named entity recognition with long short-term memory. In: *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL*, vol. 4, pp. 172–175. Association for Computational Linguistics (2003)
9. Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural Comput.* **9**(8), 1735–1780 (1997)
10. Huang, Z., Xu, W., Yu, K.: Bidirectional LSTM-CRF models for sequence tagging. arXiv preprint [arXiv:1508.01991](https://arxiv.org/abs/1508.01991) (2015)
11. Kuru, O., Can, O.A., Yuret, D.: Charner: character-level named entity recognition. In: *Proceedings of the 26th International Conference on Computational Linguistics*, pp. 911–921 (2016)
12. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: *Proceedings of the Eighteenth International Conference on Machine Learning*, vol. 1, pp. 282–289 (2001)

13. Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., Dyer, C.: Neural architectures for named entity recognition. arXiv preprint [arXiv:1603.01360](https://arxiv.org/abs/1603.01360) (2016)
14. Le, T.H., Nguyen, T.T.T., Do, T.H., Nguyen, X.T.: Named entity recognition in Vietnamese text. In: Proceedings of the Fourth International Workshop on Vietnamese Language and Speech Processing. Hanoi, Vietnam (2016)
15. Le-Hong, P.: Vietnamese named entity recognition using token regular expressions and bidirectional inference. In: Proceedings of the Fourth International Workshop on Vietnamese Language and Speech Processing. Hanoi, Vietnam (2016)
16. Hồng Phuong, L., Thi Minh Huyền, N., Roussanaly, A., Vinh, H.T.: A hybrid approach to word segmentation of Vietnamese texts. In: Martín-Vide, C., Otto, F., Fernau, H. (eds.) LATA 2008. LNCS, vol. 5196, pp. 240–249. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-88282-4_23
17. Lin, D., Wu, X.: Phrase clustering for discriminative learning. 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, vol. 2, pp. 1030–1038. Association for Computational Linguistics (2009)
18. Luo, G., Huang, X., Lin, C.-Y., Nie, Z.: Joint entity recognition and disambiguation. In: Proceedings of the 2015 Conference on Empirical Methods on Natural Language Processing, pp. 879–888. Association for Computational Linguistics (2015)
19. Ma, X., Hovy, E.: End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. arXiv preprint [arXiv:1603.01354](https://arxiv.org/abs/1603.01354) (2016)
20. Nguyen, T.C.V., Pham, T.S., Vuong, T.H., Nguyen, N.V., Tran, M.V.: Dsktlabner: nested named entity recognition in Vietnamese text. In: Proceedings of the Fourth International Workshop on Vietnamese Language and Speech Processing. Hanoi, Vietnam (2016)
21. Nguyen, T.S., Nguyen, L.M., Tran, X.C.: Vietnamese named entity recognition at VLSP 2016 evaluation campaign. In: Proceedings of the Fourth International Workshop on Vietnamese Language and Speech Processing. Hanoi, Vietnam (2016)
22. Pascanu, R., Mikolov, T., Bengio, Y.: On the difficulty of training recurrent neural networks. In: The 30th International Conference on Machine Learning, vol. 28, pp. 1310–1318. Atlanta, USA (2013)
23. Petasis, G., Petridis, S., Paliouras, G., Karkaletsis, V., Perantonis, S., Spyropoulos, C.: Symbolic and neural learning for named-entity recognition. In: Symposium on Computational Intelligence and Learning, pp. 58–66. Citeseer, Chios (2000)
24. dos Santos, C., Guimaraes, V., Niterói, R.J.: Boosting named entity recognition with neural character embeddings. In: Proceedings of NEWS 2015 the Fifth Named Entities Workshop, pp. 25–33 (2015)



Nested Named Entity Recognition Using Multilayer Recurrent Neural Networks

Truong-Son Nguyen^{1,2}(✉) and Le-Minh Nguyen¹

¹ Japan Advanced Institute of Science and Technology,
Nomi, Ishikawa 923-1292, Japan
{nguyen.son,nguyenml}@jaist.ac.jp

² University of Science, VNU-HCMC, Ho Chi Minh City, Vietnam

Abstract. Many named entities are embedded in others, but current models just only focus on recognizing entities at the top-level. In this paper, we proposed two approaches for the nested named entity recognition task by modeling this task as the multilayer sequence labeling task. Firstly, we propose a model that integrates linguistic features with a neural network to improve the performance of named entity recognition (NER) systems, then we recognize nested named entities by using a sequence of those models in which each model is responsible for predicting named entities at each layer. This approach seems to be inconvenient because we need to train many single models to predict nested named entities. In the second approach, we proposed a novel model, called multilayer recurrent neural networks, to recognize all nested entities at the same time. Experimental results on the Vietnamese data set show that the proposed models outperform previous approaches. Our model yields the state of the art results for Vietnamese with F1 scores of 92.97% at top-level and 74.74% at the nested level. For English, our NER systems also produce better performance.

Keywords: Nested named entity recognition · Deep learning
Recurrent neural networks

1 Introduction

Named entity recognition is a fundamental task in Natural Language Processing (NLP). This task aims to recognize named entities such as Person, Organization, Location, Time, and Currency in general texts or other types of text such as protein, DNA, and names of diseases in bio-medical texts.

Machine learning approaches are to be preferred to rule-based approaches in recent years by treating the task of Named Entity Recognition as the sequence labeling task or classification task.

The NER task can be solved by applying conventional machine learning algorithms independently such as Conditional Random Fields [21], Hidden Markov Model [25], Maximum Entropy [2], or by incorporating strong ML algorithms [3], or by using the classifier voting approach [21].

In another approach, [12] has used Token regular expression and Bidirectional Inference for NER task, this approach produced the best result on VLSP 2016 bench mark data set.

In NER, many entities are usually nested within another. For example, in general texts, many organization’s names contains locations inside them such as “*Bank of Washington*”, “*University of London*”, “*Japan Advanced institute of science and technology*”. Nested entities are also popular in bio-medical texts. In the GENIA corpus [19], 17% of entities are embedded in another entities. However, many current work only focus on recognizing entities at the top-level and ignored nested entities. This lack of concentration on nested entities comes from the technical reason due to an inability to model nested entities of conventional sequence models (eg. Markov model, CRF) [5].

Recently, deep learning approaches have been adapted to many NLP tasks. [9, 11] have claimed that in the NER task for English, Dutch and Spanish, deep learning models such as BI-LSTM-CRF can produce the state of the art performance without using any kinds of engineering features or additional resources. However, we think that these high performances are the results of many factors such as a good set of parameters, or a good choice of pre-trained embedding source. Consequently, without additional features, these models may not win the conventional approaches on other data sets. For example, in the VLSP 2016 campaign for Vietnamese NER¹, although [17] has used some deep learning models that achieved the state of the art the results in several benchmark NER data sets (e.g. English, Dutch, Spanish), the performance of these models on the Vietnamese NER data set is not better than some conventional approaches such as CRF [18]. Our observation is that many entities are usually follow some syntactic rules (e.g. a location is usually located after a preposition, or a named entity is usually located inside a noun phrase chunk). Such linguistic features can help to recognize named entities because and they can be obtained easily by using some NLP tools that are available in many languages. Our question is that why we don’t integrate linguistic features with deep learning models and see how contribution of these features for named entities recognition.

The designing of deep learning models is very flexible so that we can adapt the same kind of a deep learning model for different NLP tasks with small changes. For example, a recurrent neural network can be used for sequence labeling task, language generation, machine translation, sentence classification, and so on. For this reason, our motivation is to construct an unified deep learning model that can recognize nested entities and to overcome the limitation of conventional models.

Our contributions are as follows: (1) We propose a model that can integrate linguistic features with a BI-LSTM-CRF model to improve the performance of NER task; (2) We propose a sequence of single BI-LSTM-CRF models to recognize nested named entities; (3) We propose a unified model, called the multilayer BI-LSTM-CRF, that can recognize all nested entities at the same time.

Experimental results showed that using linguistic features will improve the result significantly. Especially, in the Vietnamese NER data set, deep learning

¹ http://vlsp.org.vn/evaluation_campaign_NER.

models with linguistic features as well as pre-trained embedding vectors make a significant improvement. They produce an F1 score of 92.99% at the top-level and 74.74% at the nested level and become the state of the art result in this data set. In the English NER data set, using additional linguistic features also improves the result a little in comparison to the original model.

The remainder of this paper is organized as follows. Section 2 presents briefly about task definition. Section 3 presents some background of recurrent neural networks. Section 4 describes our proposed models for the nested named entities recognition task. Section 5 presents the experimental results and some comments. Conclusion and Future works are shown in Sect. 6.

2 Nested Named Entities Recognition as Multilayer Sequence Labeling Task

We can model the task of recognizing nested named entities as a multi-layer sequence labeling task by organizing nested entities in n different layers. The value of n is the maximum nested level in the training data set. The input is a sequence of words or tokens, and the output is n sequences of tags in which each sequence represent named entities that are recognized at each level. We can use IOB notation to represent named entities [6] in which B-, I- tags are used to mark the beginning and inside of name entities, other tokens that are outside of any named entities are marked by the tag O. For example, in Fig. 1, the input sentence with three levels of entities (one top-level and two nested levels) is modeled as the multilayer sequence labeling task with three layers.

Input	Edinburgh	University	Library	was	located	at	George	Square
Output								
Layer 1	B-ORG	I-ORG	I-ORG	O	O	O	B-LOC	I-LOC
Layer 2	B-ORG	I-ORG	O	O	O	O	B-PER	O
Layer 3	B-LOC	O	O	O	O	O	O	O

Fig. 1. Nested named entities as a multilayer sequence labeling task

3 Recurrent Neural Network for Sequence Labeling Task

3.1 Long Short Term Memory (LSTM)

Long short term memory networks (LSTM) [8] is a variant of recurrent neural networks (RNNs), a kind of neural networks that can operate sequential data, that is suitable for solving sequence labeling tasks.

An RNN [4] showed in Fig. 2, has an input layer \mathbf{x} , a hidden layer \mathbf{h} and an output layer \mathbf{y} . In the sequence labeling task, \mathbf{x} represents input embedding vectors of tokens $\mathbf{x} = (x_1, x_2, \dots, x_n)$ and \mathbf{y} present tags $\mathbf{y} = (y_1, y_2, \dots, y_n)$. If we consider a sequence is a kind of time series data, each embedding vector x_t

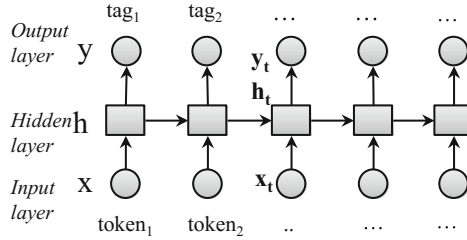


Fig. 2. Recurrent neural networks

represents features of **token_t** at time **t**. They could be one-hot-encoding vector for word feature, dense vector features or sparse vector features. The size of the input layer is the dimension of these features vectors. A vector y_t of the output layer of size L represents a probability distribution over all tags at the time **t**. The dimension L of this vector is the number of distinct tags and its values are usually achieved by using a softmax activation function $\mathbf{y}_t = \text{softmax}(\mathbf{V}\mathbf{h}_t)$. The value of h_t can be computed depend on which kind of RNN network chosen. Below is the implementation in [11] that we also use in our LSTM-based models:

$$\begin{aligned}
 \mathbf{i}_t &= \sigma(\mathbf{W}_{xi}\mathbf{x}_t + \mathbf{W}_{hi}\mathbf{h}_{t-1} + \mathbf{W}_{ci}\mathbf{c}_{t-1} + \mathbf{b}_i) \\
 \mathbf{c}_t &= (1 - \mathbf{i}_t) \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tanh(\mathbf{W}_{xc}\mathbf{x}_t + \mathbf{W}_{hc}\mathbf{h}_{t-1} + \mathbf{b}_c) \\
 \mathbf{o}_t &= \sigma(\mathbf{W}_{xo}\mathbf{x}_t + \mathbf{W}_{ho}\mathbf{h}_{t-1} + \mathbf{W}_{co}\mathbf{c}_t + \mathbf{b}_o) \\
 \mathbf{h}_t &= \mathbf{o}_t \tanh(\mathbf{c}_t)
 \end{aligned}$$

where σ is the element-wise sigmoid function, and \odot is the element-wise product. \mathbf{W}_{ij} matrices are connection weights, \mathbf{i}_t and \mathbf{o}_t are gates that control the proportion of input to give to the memory cell \mathbf{c}_t and the memory cell \mathbf{c}_t to the output, respectively.

3.2 Bidirectional Long Short Term Memory (BI-LSTM) and BI-LSTM-CRF

Figure 3 shows the architecture of a BI-LSTM network [7] in which the value of hidden vector is the concatenation between the forward and backward hidden vectors ($\mathbf{h}_t = [\vec{\mathbf{h}}_t, \overleftarrow{\mathbf{h}}_t]$). In comparison with a single direction LSTM, an BI-LSTM can use the information from the both sides to predict the label of the token at each time step.

BI-LSTM-CRF is the combination between BI-LSTM and Conditional Random Fields, a strong algorithm for sequence labeling tasks which is more effective than Hidden Markov models as well as Maximum Entropy [10]. In an LSTM or a BI-LSTM model, the tagging decision at the output layer is made independently using a *softmax* activation function. That means the final tagging decision of a token does not depend on the tagging decision of others. Therefore, adding a CRF layer into an LSTM or a BI-LSTM model equips the model the ability to

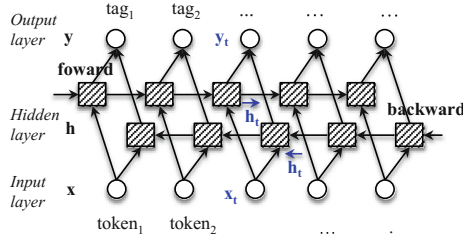


Fig. 3. Bidirectional long short-term memory

learn the best sequence of tags that maximizes the log probability of the output tag sequence. This model are described details in [9, 11, 24]. BI-LSTM-CRF is very successful for the Named entities recognition task. They produced the state of the art results on several NER benchmark data sets without using any features. Our proposed models in this paper are based on the BI-LSTM-CRF model and are described in details in Sect. 4.

Training neural networks. We can train all neural network models using the back propagation method, in which parameters/weights of the neural networks will be updated through time to optimize the objective function (e.g. the cross entropy loss function) by using some popular methods such as Stochastic gradient descent (SGD). Besides, a dropout technique can be applied to avoid the over-fitting problem [20].

Initializing of word embedding vectors. Using Pre-trained word embedding vectors is a way to improve the performance of the system. These vectors can be initialized randomly or from pre-trained embedding sources. We can train word embedding vector ourselves using word2vec [16] or some of its variation such as skip-n-gram [14]. If word embedding vectors are initialized randomly, these embedding vectors can be optimized during the training process.

4 Proposed Models

4.1 Single BI-LSTM-CRF with Features

This model is a modification of BI-LSTM-CRF proposed by [11]. Figure 4 shows the architecture of our BI-LSTM-CRF. Suppose that each word/token in the input sequence has some features such as Part-of-speech, Chunk beside the head word. Based on the original model, with each feature, we add an embedding layer that represented that feature. For example, with a word at the time t (e.g., “Library”), we use a word embedding representation of this word, an embedding vector represents its POS feature, an embedding vector represents its Chunk feature and so on. All embedding vectors are obtained from several lookup tables which map from a categorical value (e.g. word, POS feature) into an embedding vector. Then, the input vector x_t is obtained by concatenating all embedding vectors of that word. While the word embedding can be initialized randomly

or from a pre-trained embedding source, the embedding vectors of features are initialized randomly. The computation of the hidden layer and output layer in our model is the same as the original model.

The objective of training procedure is to find the best weights of deep learning models that minimize the loss function which measures the difference between the output tags of the model and ground truth tags. We use the same loss function as [11].

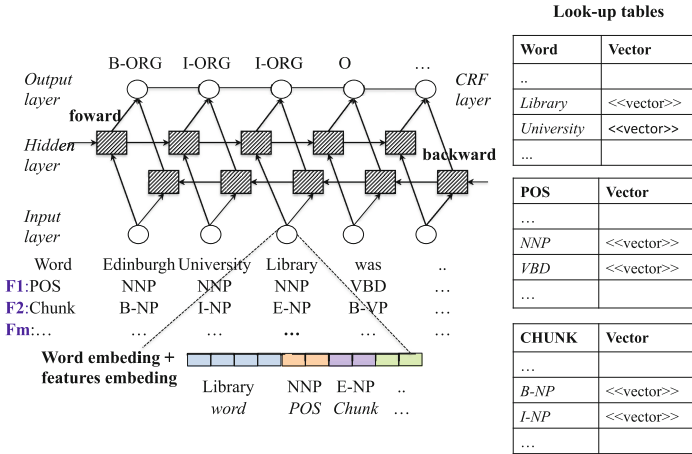


Fig. 4. BI-LSTM-CRF with feature embedding layers

4.2 Sequence of Single BI-LSTM-CRF Model

By modeling the nested NER as a multilayer sequence labeling task, we propose the sequence of single BI-LSTM-CRF models (described in Sect. 4.1) to recognize tags at each layer. In the training phase, we first determine n as the number of layers in the training corpus. Then we train n BI-LSTM-CRF models separately. In order to train model i^{th} , we use word embedding, linguistic features and ground truth tags of layer 1 to $i - 1$ as additional features. In the prediction phase, to predict tags of layer i , we must use trained models to predict tags of previous layers (1 to $i - 1$) then use these tags as features to predict tags of layer i . Finally, the output of the prediction phase is the union of tags of all single models.

4.3 Multilayer BI-LSTM-CRF Model

Using the sequence of single models to recognize nested entities is inconvenient for training and prediction process because we have to train each model to recognize the label at each nested level. For the prediction phase, we have to recognize the label of the low layers then using these labels as features for predicting labels of higher layers. To overcome this inconvenience, we proposed a unified model

that make the training and prediction process simple because we only need to train one single model to predict labels of all nested levels at the same time.

The architecture of multilayer BI-LSTM-CRF model is illustrated in Fig. 5. This model consists of n child BI-LSTM-CRF layers in which the layer i^{th} is responsible for predicting labels at that layer and computes additional feature vectors for higher layers (layer $i + 1$ to n).

For example, the BI-LSTM-CRF model of the first layer will take word embedding and feature embedding vectors of all words in a sentence as the input. Then it will compute output vector that represents the distribution over all labels of each word. These vectors will be used as the additional features for higher layers (layer 2 to n) as well as to calculate the label for this layer.

The loss function of the whole model is the sum of the loss of all layers (Eq. 1). Loss function at each layer is the lost function of each single BI-LSTM-CRF model presented in Sect. 4.1.

$$loss = \sum_{i=1}^n loss_i \tag{1}$$

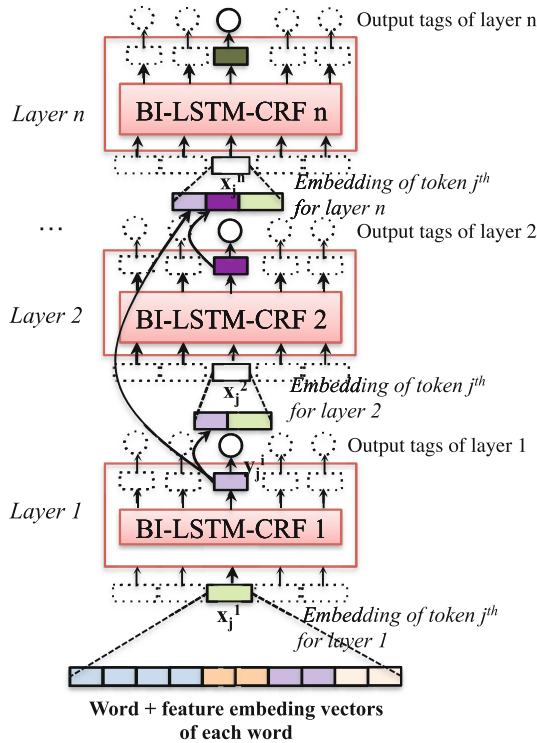


Fig. 5. Multilayer BI-LSTM-CRF architecture

5 Experiments

5.1 Data Sets, Evaluation Method, and Experimental Settings

We evaluate proposed models on two benchmark data sets including the Vietnamese NER data set obtained from the VLSP community [18] and the English NER data set obtained from CoNLL 2003 Shared Task [23]. Both of POS and Chunk tags are available in these two data sets. Table 1 shows the statistics of Vietnamese NER data set. The Vietnamese NER data set has two levels including the top-level and the nested level. Although the English NER has only one level, it has still been used to evaluate the effectiveness of linguistic features.

Table 1. The statistics of the Vietnamese NER data set

Type	Top-level			Nested level			Percent
	Train	Dev	Test	Train	Dev	Test	
PER	5803	703	1294	7	0	7	46.0%
LOC	5057	532	1377	382	51	100	44.2%
ORG	939	151	274	1	0	0	8.0%
MISC	222	33	49	0	0	0	1.8%
# total of NE	12021	1419	2994	390	51	107	

NER systems are evaluated at *Precision*, *Recall*, and $F_{\beta=1}$ scores using a third-party evaluation tool provided by CoNLL 2000 Shared task [22]. For each experiment, we trained the model after 120 epochs and choose the one that produced the best result on the validation data set. Then, we use that model to evaluate the performance of the model on the official test set. We do not use the test set when choosing the best model. To train neural networks, we also used the back-propagation method with the stochastic gradient descend method with the learning rate at 0.005. Moreover, we set the dropout rate to 0.5 to avoid the over-fitting problem.

5.2 Tuning Parameters

There are many parameters need to be tuned in the BI-LSTM-CRF model such as the size of word embedding, POS-embedding, and Chunk-embedding; the number of nodes in hidden layers; dropout rate and so on. Because tuning all parameters is time-consuming, we just only focus on tuning some parameters that related our proposed model including the size of POS embedding vector, chunk-embedding vector. The validation data set will be used to tune the parameters.

Pre-trained word embedding representation. Pre-trained word embedding representation is very important for deep-learning models. In many previous experiments for NLP tasks, using pre-trained word embedding will help NLP system

achieve the better results. To obtain the pre-trained embedding vectors for Vietnamese, We have crawled 500k articles (14.6 million sentences and 263.3 million words, respectively) from Wikipedia and some popular news websites in Vietnamese such as <http://vnexpress.net> and <http://tuoitre.com.vn>. These websites contain many named entities and that may be a good embedding source for this task. After crawling news article, we extract the raw content and recognize word segmentation using vnTokenizer tool [13]. Then, we train the word embedding vectors use word2vec tool [16].

Table 2 shows some interesting results after training word embedding representation. Our observation shows that the top nearest with an entity are usually entities in the same type. For examples, given an entity such as Obama (PERSON), the top nearest with this entity also referred to another person such as Putin, Barack_Obama, Kerry, Hagel, Abe, Vladimir_Putin, Hollande, Biden, Merkel, Hillary_Clinton or LOCATION such as Nhà_Trắng (White House). Based on these examples, we expect that word embedding representation that trained from news website is a valuable resource for discovering new entities that do not appear in the training corpus.

Table 2. Top nearest words of some named entities based on word embedding representation trained from Vietnamese news websites

Word	Top 10 nearest words
Obama PERSON	Putin, Barack_Obama, Kerry, Nhà_Trắng, Hagel, Abe, Vladimir_Putin, Hollande, Biden, Merkel, Hillary_Clinton
Huế LOCATION	Gia_Định, Qui_Nhon, Dục_Thanh, Kontum, Chợ_Quán, Quảng_Nam, La_Vang, Vạn_Hạnh, Quảng_Tri
Vietcombank ORGANIZATION	ACB, Techcombank, Eximbank, Vietinbank, VietinBank, BIDV, Maritime_Bank, HOSE,VPBank,Sacombank

For the task NER in English, we choose pre-trained word embedding representation from [15], an adaptation of word2vec for syntax problems. Experimental results show that it is better than word2vec or glove. This pre-trained embedding source is also used in [11].

5.3 Results and Discussions

Table 3 shows the experimental results on Vietnamese NER data set. In comparison to the original model, our proposed model showed a significant improvement. Between two linguistic features, The Chunk feature is better than POS and the combination between them shows the best result. For example, from model BI-LSTM-CRF 1 to 4, using POS feature improves +3.54%, using Chunk feature

Table 3. Performance comparison between different feature sets at the top-level. Model 1 and 5 is the original BI-LSTM-CRF model.

Model	POS dim = 30	CHUNK dim = 30	Pre- trained	F1 %		
BI-LSTM-CRF	1			82.9	baseline1	
	2	X		86.44	+3.54%	
	3		X	89.77	+6.87%	
	4	X	X	90.27	+7.37%	
	5			X	86.84	baseline2
	6	X		X	88.66	+1.82%
	7		X	X	91.79	+4.95%
	8	X	X	X	92.97	+6.13%
Token regular expressions and bidirectional inference (Le 2016)				88.78		
CRF with POS and Chunk				87.06		

improves +6.87% and using both of them improves +7.37% than *baseline1*. This phenomenon is the same with models from BI-LSTM-CRF 5 to 8. Besides, our models outperform other systems on this data set. In comparison with the best system in VLSP 2016 shared task [12], our models not only improve the result significantly (4.2% of $F_{\beta=1}$ score) but also our models focus on both of recognizing named entities the top-level and the nested level.

Beside that, pre-trained embedding representation play an important role for NER systems. Models with pre-trained embedding representation always produce the better result than the one without it. In addition, tuning the size

Table 4. Performance comparison between different configuration of POS dim, CHUNK dim at the top-level named entities. Symbol * indicates the result the test set when have the best result on the validation set

	POS dim	CHUNK dim	Pre-trained embedding	
			Size = 100	Size = 200
BI-LSTM-CRF	0	0	86.84	86.6
	0	30	91.79	91.25
	0	50	91.76	91.97
	30	0	88.66	87.42
	30	30	92.97*	91.97
	30	50	92.56	91.51
	50	0	88.37	86.22
	50	30	92.57	91.59*
	50	50	93.00	92.09

Table 5. Performance comparison between the sequence of BI-LSTM-CRF, the multilayer BI-LSTM-CRF, Sequence of CRF (using CRF++)

Pos	Chunk	Sequence of BI-LSTM-CRF			Multilayer BI-LSTM-CRF			Sequence of CRF++				
		Top level (1)	Nested level (2)	All levels (3)	Top level (4)	Nested level (5)	All levels (6)	Top level (7)	Nested level (8)	All levels (9)	(6)–(3)	(6)–(9)
		86.84	61.45	86.09	87.28	55.62	86.31	65.98	29.69	65.02	+0.22%	+21.29%
X		88.66	62.15	87.90	87.54	52.87	86.55	81.62	42.11	80.64	−1.35%	+5.91%
	X	91.79	70.18	91.18	92.52	70.65	91.85	78.46	59.87	77.91	+0.67%	+13.94%
X	X	92.97	74.35	92.39	92.99	74.74	92.43	87.06	62.98	86.38	+0.04%	+6.05%

of word embedding layer as well as feature embedding layer does not affect the results much (see Table 4).

The sequence of BI-LSTM-CRF models and the multilayer BI-LSTM-CRF model shows the very close results (see Table 5). The multilayer show the best result in which F1 scores at top-level and nested level are 92.99% and 74.74 %. Moreover, the performance at the nested level decreases sharply if the performance at the top-level decrease (see Table 6). That shows a strong relationship between entities in different levels.

Table 7 shows the results in the English data set. Using POS tags as additional features help to improve the performance a little compared to the original model (see Table 7). However, chunk features do not produce a good result as our expectation. That result is very closed to the state of the art result which reported by [1]. In their work, they used the Joint Entity Recognition and Linking to jointly model NER and linking tasks and used a lot of hand-engineered features including spelling features, WordNet clusters, Brown clusters, POS tags, chunks tags, as well as stemming and external knowledge bases like Freebase and Wikipedia.

Proposed models are implemented using Theano library with Python programming language².

Table 6. Results of Sequence of BI-LSTM-CRF models at the nested-level in two cases: (a) if entities at the top-level are absolutely correct (b) entities at the top-level are predicted use our trained models

POS dim = 30	CHUNK dim = 30	F1 % (a)	F1 % (b)
		79.14	61.45
X		81.91	62.15
	X	79.57	70.18
X	X	79.80	74.35

² The systems are available at <https://github.com/ntson2002/lstm-crf-tagging>.

Table 7. Experimental results on the English NER data set

Model	F1
Luo et al. (2015) + gazettee + linking	91.2
Lample et al. (2015) BI-LSTM-CRF	90.94
<i>Our model:</i>	
* BI-LSTM-CRF + Chunk	90.68
* BI-LSTM-CRF + POS	91.19
* BI-LSTM-CRF + Chunk + POS	90.88

6 Conclusion and Future Works

In this work, we firstly proposed an integration between linguistic features into a BI-LSTM-CRF model. We exploit the contribution of two linguistic features including POS and chunk tags. Experiments conducted on both English and Vietnamese NER data sets showed that linguistic features play an important role to improve the result of the NER task when incorporate them with deep learning models. Our implementation can expand to use other kinds of engineering features such as gazetteers or hand-craft rules without changing anything. Secondly, we introduce the sequence of BI-LSTM-CRF models and the multi-layer BI-LSTM-CRF model to recognize nested named entities. The multilayer BI-LSTM-CRF model produces the competitive performance as the sequence of BI-LSTM-CRF and its design is very convenient because we only need to train one model to recognize all entities at all nested levels. In future, we will conduct experiments on other nested entities data sets such as GENIA corpus or exploit more features to improve the performance of NER systems.

Acknowledgment. This work was supported by JSPS KAKENHI Grant number JP15K16048.

References

1. Joint Named Entity Recognition and Disambiguation, September 2015. <https://www.microsoft.com/en-us/research/publication/joint-named-entity-recognition-disambiguation/>
2. Borthwick, A.: A maximum entropy approach to named entity recognition. Ph.D. thesis, Citeseer (1999)
3. Chieu, H.L., Ng, H.T.: Named entity recognition with a maximum entropy approach. In: Daelemans, W., Osborne, M. (eds.) Proceedings of CoNLL-2003, Edmonton, Canada, pp. 160–163 (2003)
4. Elman, J.L.: Finding structure in time. *Cogn. Sci.* **14**(2), 179–211 (1990)
5. Finkel, J.R., Manning, C.D.: Nested named entity recognition. In: EMNLP 2009, pp. 141–150. Association for Computational Linguistics (2009)

6. Florian, R., Ittycheriah, A., Jing, H., Zhang, T.: Named entity recognition through classifier combination. In: Daelemans, W., Osborne, M. (eds.) *Proceedings of CoNLL-2003*, Edmonton, Canada, pp. 168–171 (2003)
7. Graves, A., Mohamed, A.R., Hinton, G.: Speech recognition with deep recurrent neural networks. In: *2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6645–6649. IEEE (2013)
8. Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural Comput.* **9**(8), 1735–1780 (1997)
9. Huang, Z., Xu, W., Yu, K.: Bidirectional LSTM-CRF models for sequence tagging. *arXiv preprint [arXiv:1508.01991](https://arxiv.org/abs/1508.01991)* (2015)
10. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: *ICML*, vol. 1, pp. 282–289 (2001)
11. Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., Dyer, C.: Neural architectures for named entity recognition. *arXiv preprint [arXiv:1603.01360](https://arxiv.org/abs/1603.01360)* (2016)
12. Le-Hong, P.: Vietnamese named entity recognition using token regular expressions and bidirectional inference. *arXiv preprint [arXiv:1610.05652](https://arxiv.org/abs/1610.05652)* (2016)
13. Le-Hong, P., Roussanaly, A., Nguyen, T.H., Rossignol, M.: An empirical study of maximum entropy approach for part-of-speech tagging of Vietnamese texts. In: *Traitement Automatique des Langues Naturelles-TALN 2010*, p. 12 (2010)
14. Ling, W., Chu-Cheng, L., Tsvetkov, Y., Amir, S.: Not all contexts are created equal: better word representations with variable attention (2015)
15. Ling, W., Dyer, C., Black, A., Trancoso, I.: Two/too simple adaptations of word2vec for syntax problems. In: *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics (2015)
16. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: *Advances in Neural Information Processing Systems*, pp. 3111–3119 (2013)
17. Nguyen, T.S., Tran, X., Chien, N.L.M.: Vietnamese named entity recognition @ VLSP 2016 evaluation campaign. In: *The Fourth International Workshop on Vietnamese Language and Speech Processing*, pp. 18–23 (2016)
18. Nguyen, T.M.H., Vu, X.L.: Report on named-entity recognition evaluation campaign: data and systems. In: *The Fourth International Workshop on Vietnamese Language and Speech Processing*, pp. 1–5 (2016)
19. Ohta, T., Tateisi, Y., Kim, J.D.: The GENIA corpus: an annotated research abstract corpus in molecular biology domain. In: *Proceedings of the Second International Conference on Human Language Technology Research*, pp. 82–86. Morgan Kaufmann Publishers Inc. (2002)
20. Srivastava, N., Hinton, G.E., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **15**(1), 1929–1958 (2014)
21. Thao, P.T.X., Tri, T.Q., Dien, D., Collier, N.: Named entity recognition in Vietnamese using classifier voting. *ACM Trans. Asian Lang. Inf. Process. (TALIP)* **6**(4), 3 (2007)
22. Sang, E.F.T.K., Buchholz, S.: Introduction to the CoNLL-2000 shared task: chunking. In: *Proceedings of the 2nd Workshop on Learning Language in Logic and the 4th Conference on Computational Natural Language Learning*, vol. 7, pp. 127–132. Association for Computational Linguistics (2000)

23. Sang, E.F.T.K, De Meulder, F.: Introduction to the CoNLL-2003 shared task: language-independent named entity recognition. In: Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003, vol. 4, pp. 142–147. Association for Computational Linguistics (2003)
24. Wang, P., Qian, Y., Soong, F., He, L., Zhao, H.: A unified tagging solution: bidirectional LSTM recurrent neural network with word embedding. arXiv preprint [arXiv:1511.00215](https://arxiv.org/abs/1511.00215) (2015)
25. Zhou, G., Su, J.: Named entity recognition using an HMM-based chunk tagger. In: proceedings of the 40th Annual Meeting on Association for Computational Linguistics, pp. 473–480. Association for Computational Linguistics (2002)

Text Summarization



Deletion-Based Sentence Compression Using Bi-enc-dec LSTM

Dac-Viet Lai¹(✉), Nguyen Truong Son^{1,2}, and Nguyen Le Minh¹

¹ Japan Advanced Institute of Science and Technology, Nomi, Ishikawa, Japan
{vietld,nguyen.son,nguyenml}@jaist.ac.jp

² University of Science, VNU-HCMC, Ho Chi Minh City, Vietnam
<https://nguyenlab.github.io/>

Abstract. We propose a combined model of enhanced Bidirectional Long Short Term Memory (Bi-LSTM) and well-known classifiers such as Conditional Random Field (CRF) and Support Vector Machine (SVM) for compressing sentence, in which LSTM network works as a feature extractor. The task is to classify words into two categories: to be retained or to be removed. Facing the lack of reliable feature generating techniques in many languages, we employ the obtainable word embedding as the exclusive feature. Our models are trained and evaluated on public English and Vietnamese data sets, showing their state-of-the-art performance.

Keywords: Text summarization · Sentence compression
Bidirectional LSTM · Sequence to sequence
Conditional random field · Support vector machine

1 Introduction

In natural language processing, sentence compression is defined as generating a shorter sentence from a long complicated sentence. Plenty of compression systems have been introduced, they can be categorized into *extractive* and *abstractive* methods. In abstractive approaches [2, 4, 22], an original sentence will be shortened by introducing some paraphrased words. Whereas, in extractive methods [6, 7, 13, 18] words are assessed to be deleted or retained in the same order as those in the given sentence. The goal of this task is to keep the significant information in a short generated sentence while remaining grammatically correct.

Most existing approaches take advantages of the affluent feature set in order to abate the possibility of grammatical mistakes. Syntactic features were used in some previous works [3, 18]. Other researches introduced pruning tree methods in which subtrees were pruned from parsed trees such as syntactic tree and dependency tree [4, 8, 21, 26]. Semantic information and word relation were also employed to figure out how to reorder and paraphrase sentences [15, 16]. Those methods require a huge linguistic feature set and a reliable tree parser which

are not available in many languages. By the development of deep learning, distributed word representation has been introduced as new potential features [19] so that this problem can be effectively solved in recent works [6, 12, 24].

The lack of features in many languages poses a challenge to develop a new solution. We settle the shortage of features by adapting the neural network taking available word embedding features as unique input. We intensify the well-known bidirectional LSTM model with forward and backward encoders. Our models also prove that reputable classifiers such as support vector machine and conditional random field are feasible to supply a better performance than a linear classifier. The main contributions of this paper are:

- We solved summarization problem by sequence tagging model.
- We equipped the bidirectional LSTM with additional encoders.
- We investigate the replacement of the linear classifier layer with conventional classifiers.
- We made the system available to the public as a web application¹

The rest of the paper is organized as follows: Sect. 2 introduces other relevant systems using deep neural network; Sect. 3 describes the adaptation of Bi-LSTM-CRF [11] and proposed models; Sect. 4 shows the specific settings and experiments; we present the results and discuss the strengths and weaknesses of our models in Sect. 5; Sect. 6 states our conclusion and unsolved issues.

2 Related Works

In this paper, the problem is defined as: given an input sentence, the system must decide which words should be kept and which words should be deleted from the original sentence. For each word, we have two options to decide, therefore, each option may be stick with a given label. Overall, we transform a sequence of tokens into a sequence of binary labels with the same length.

Recurrent neural network (RNN) has shown its power in solving many natural language processing problems such as parsing [5], speech recognition [9], machine translation [17], word representing [19, 20], text summarization [1, 2]. For sentence compression task, LSTM - the most well-known version of RNN - significantly contributes to the performance of the flexible compression systems [6, 12, 24] that require a short list of available features while boosting the efficiency.

Sentence compression can be considered as a translation task [6], in which three-layer LSTM stack is the core translating text to sequence of three labels (0, 1 and EOS (end of sentence)). The one-hot vector resulted from previous step and the pre-trained word embedding vector are used as the main features. They also show the good contribution of POS-tag in making decision. Currently, a customized attention mechanism called t-attention has been proven to effectively obtain conditional context [24]. In t-attention, during decoding stage, the global

¹ <http://150.65.242.97/sum/en/>.

context attention technique is replaced by a local attention mechanism which refers to the hidden state of the corresponding step in the encoding stage. In order to enrich the feature set, eye-tracking information and combinatory categorial grammar (CCG) are integrated. They trained a multi-task learning model [12] which targets at sentence compression and CCG super tagging. Though multi-target learning is helpful to learn higher abstract level, the additional features demand gaze data and CCG parser, hence breaks the root idea of utilizing cheap available features.

3 Models

In this paper, we model the task as a sequence tagging. The model is trained to maximize the probability of generating correct compressed sentence. Given a pair of original sentence and its compressed version (X, Y) , the model learns its parameters θ to optimize the following problem:

$$\theta' = \operatorname{argmax} \sum \log p(Y|X; \theta) \quad (1)$$

3.1 Bi-LSTM-CRF Model

Bi-LSTM-CRF model [11], has shown many advantages for named entities recognition task. Without using engineering features, this model has produced the state of the art result on several benchmark data sets [14]. Given a sentence S of length T , the Bi-LSTM network delivers a matrix of scores $[f_{\theta}(S)]$. $[f_{\theta}(S)]_{i,t}$ denotes the score of tag i -th for the t -th word in the sentence S . A learned transition score $[A]_{i,j}$ indicates the score of the transition from state i -th to state j -th for a pair of adjacent words (Fig. 1). Then the score of a sequence of labels $[l]$ with respect to the sentence S is calculated by:

$$\operatorname{score}(S, [l], \theta, [A]) = \sum_{t=1}^T ([A]_{[l]_{t-1}, [l]_t} + [f_{\theta}(S)]_{[l]_t, t})$$

As the result, Bi-LSTM-CRF has the following characteristics. Firstly, it is based on LSTM [10] - a variant of recurrent neural networks (RNNs) that is able to capture the long constraints and solve the gradient vanishing and exploding problem.

Secondly, the forward LSTM can predict the label of the current time based on the previous information. For example, in sequence labeling problems, a LSTM network predicts the label of a token based on the information learn from the preceding tokens. However, the relation of words in a sentence is two-way dependency. That means the information of the current token may be affected by both the previous and the next tokens. Therefore, a Bi-LSTM allows the ability to learn from the past and future information to predict the label at the current time.

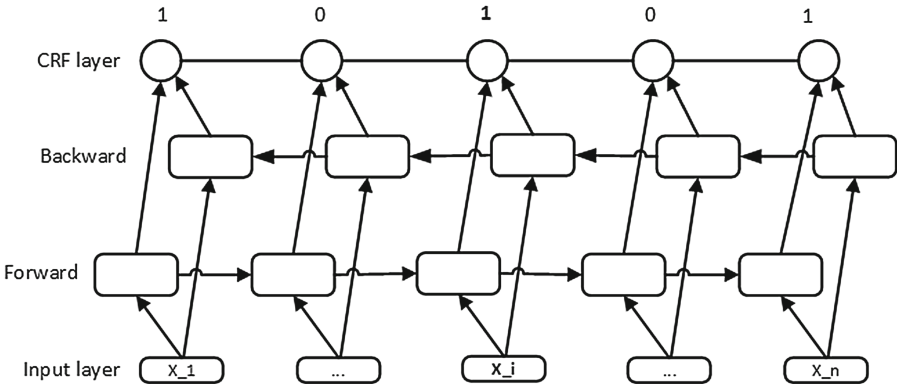


Fig. 1. BI-LSTM-CRF model

Thirdly, without CRF-layer, the labels are generated independently in a LSTM model or a Bi-LSTM model. That means the final tagging decision of a token does not depend on the others. Therefore, adding a CRF layer into a LSTM model or a Bi-LSTM model equips them with the ability to decide the best sequence of tags that maximize the log probability of the output tag sequence.

3.2 Bi-encoder-decoder Model

In sequence tagging, each LSTM model has their own strengths. Bi-LSTM [11] overcomes the lack of information in both directions while leaving words at the beginning and the end misunderstood. The reason is the hidden state in one of the direction is strongly affected by the initialization of cell's parameters and states. In sequence to sequence model [23], the decoder performs well because the whole information of the input is captured by the encoder then passed to decoder through hidden states (Fig. 2).

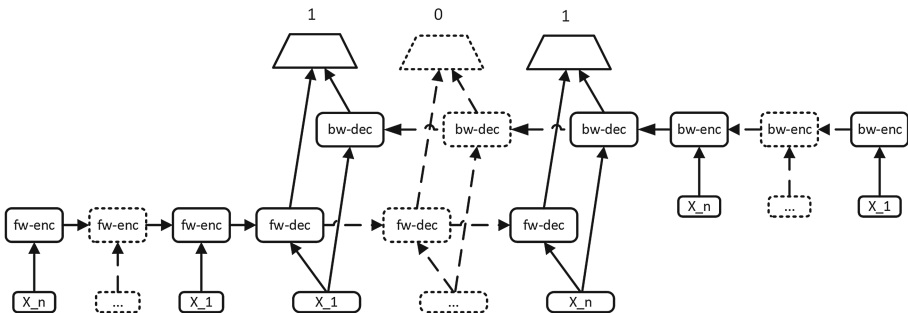


Fig. 2. The bi-encoder-decoder model

Merging these two models, our model mainly enhances the bidirectional LSTM and the sequence to sequence model so that they can both avoid the domination of variable initialization and capture information from both direction. Specifically, the given input sentence (x_1, x_2, \dots, x_n) is passed though three phases as follows.

- First, the encoders capture the general information of the sentence in separated backward and forward networks.
- After initializing the hidden state by the encoders, the bidirectional LSTM network works as the decoder extracting features.
- Finally, a classifier located on top of the decoder concatenates features from both directions, then labels the tokens at regarding time step as **0** or **1**. Unlike [23], we move the **EOS** token from decoder parts to the encoder parts. This small modification ensures the same inputs for two LSTM cells in both sides and keeps the balance of the two chains of network in bidirectional model. Mathematically, we decompose the Eq. 1 as follows:

$$p(Y|X; \theta) = \prod_{t=1}^T p(Y_t|X; \theta)$$

We employ the original idea of LSTM cell from [10] accepting x_t, h_t, m_t as input, control state and memory state at timestep t , respectively. Then given a sequence of input, the chain of recurrent computation process follows:

$$\begin{aligned} i_t &= \text{sigmoid}(\theta_1 x_t + \theta_2 h_{t-1}) \\ i'_t &= \text{tanh}(\theta_3 x_t + \theta_4 h_{t-1}) \\ f_t &= \text{sigmoid}(\theta_5 x_t + \theta_6 h_{t-1}) \\ o_t &= \text{sigmoid}(\theta_7 x_t + \theta_8 h_{t-1}) \\ m_t &= m_{t-1} \cdot f_t + i_{t-1} \cdot i'_t \\ h_t &= m_t \cdot o_t \end{aligned} \tag{2}$$

In which, \cdot is the element-wise product, θ_i is the parameters of the cells, h_0 is initialized as a *zero* matrix.

Similar to [6] we use the three-layer stacked LSTM model. This model, when integrated with dropout layers, helps capture high-level meaning of words and abates the over-fitting effect. The trainable linear layer works as the classifier making up the bi-encoder-decoder model. This layer, taking the features from decoders, generates two labels: **0** indicates that the corresponding token will be deleted; **1** indicates that the token will be maintained in the compressed sentence.

The computing process of the first two phases can be denoted as an extractor described in the pseudocode 1. Given a sentence *sent*, the backward encoder *Bw_enc* and forward encoder *Fw_enc* compute the hidden states from the sentence and its reversed version. After that, the hidden states are put in the Bi-LSTM network as the initial hidden states.

Algorithm 1. Extracting features from given input sequence.

```

procedure EXTRACT(Fw_Enc, Bw_Enc, BiLSTM, sent)
  fw_state  $\leftarrow$  Fw_Enc(REVERSE(sent))
  bw_state  $\leftarrow$  Bw_Enc(sent)
  features  $\leftarrow$  BiLSTM(fw_state, bw_state, sent)
return features
end procedure

```

3.3 The Combined Bi-encoder-decoder Model

Despite linear classifier being well-known for solving multi-class problem, we propose to replace it with other classifiers. We choose SVM to be a potential classifier because it was originally designed for two-class classification. We attempt to utilize trainable linear layer, namely **Bi-enc-dec** model, against the combination model called **Bi-enc-dec-SVM**. The linear layer is used during the extraction process. After training the network, the output of the highest LSTM layer is collected as features. The output of each time step is examined as a sample for training SVM. It is essential to clarify that thanks to the separated training process, the error does not pass through the SVM layer to the neural network.

4 Experiment

4.1 Data Sets

We use data set for training and evaluating in two languages, English and Vietnamese. Each data set is splitted into 10 folds, in which the ratio for training, validation and testing is 8:1:1. Table 1 shows the characteristics of the data sets.

Table 1. Data set statistics.

	English	Vietnamese
#Pairs	10,000	2,929
Vocab	25,659	6,525
Max length	210	151
Average length	26.66	25.15
Compression rate	0.41	0.65
Train vocab coverage	0.94	0.95

The English data set [6] contains 10000 pairs of original and compressed sentences. We conduct the experiment on the two well-known distributed word

representations: word2vec [19] and GloVe [20]. The pre-trained 100-dimensional GloVe² word embedding is generated from a 6-billion-token corpus. We generate word2vec vectors from the “one billion word modeling benchmark corpus”³ with window size of 8 and 256-dimensional vector. This large corpora reduce the probability of unknown token during training process. All pairs of original and compressed sentences are tokenized by Stanford word tokenizer⁴. In case of unrecognized tokens, we try to use their lemmatized token instead. At worst, unknown tokens will be replaced by an “UNK” (unknown) token, which will be represented by a uniformed random vector. We also utilize end-of-sentence “EOS” token to mark the end of each sentence at the end of encoding progress.

For Vietnamese, the “Written Vietnamese Cluster Corpus for Document Summarization” (WVCCDS) [25] is a manual compressed data set. Corresponding to each original sentence, there are three compressed version independently created by three annotators. We choose the version with compression rate (CRate - the average percentage of the number of retained tokens) nearest to the average of the whole dataset. We collect a corpus of 500.000 news from Vietnamese online news⁵ in order to build the word embedding. This corpus contains above 14 million sentences and 263 million tokenized words.

4.2 Evaluation

Similar to [6], our models are measured on F1-score and compression rate. The system performs well in term of compression rate if its output CRate is close to that of human-generated sentence. Additionally, we calculate the ROUGE score in order to better measure the efficiency in the view of summarization task.

Targeting to compare the performance of linear layer, CRF layer and SVM layer, three models: **Bi-enc-dec**, **Bi-enc-dec-CRF** and **Bi-enc-dec-SVM** are investigated. These models share the same encoder and decoder. A SVM classifier is trained, replacing the linear layer. The learning rate is set at 0.007 to optimize the neural network. Three stacked LSTM layers of the encoder and the decoder have the number of units equal to the input vector size. The models are not optimized through a predefined number of epochs, instead, the training process stops whenever the lost value in the validation set increases.

4.3 Baseline and Settings

For the English dataset, the sequence to sequence model [6] was trained with 1.8 million sentences, therefore, to be fair, we re-implement, then train the model

² <http://nlp.stanford.edu/data/glove.6B.zip>.

³ <https://code.google.com/archive/p/word2vec/>.

⁴ <http://nlp.stanford.edu/software/tokenizer.shtml>.

⁵ <http://vnexpress.net>.

with the same training set as follow. We built a three-layer stacked LSTM sequence to sequence model. At the decoder, the output of the preceding step is fed as 3-dimensional one hot vector. The model parameters such as number of hidden units, learning rate, and so on are kept the same as described in [6]. The result of other reference models are picked from the authors' papers.

We train the four proposed models: the **Bi-LSTM-CRF** and its enhanced version **Bi-enc-dec-CRF**, **Bi-enc-dec** with bidirectional encoder-decoder and the combined model **Bi-enc-dec-SVM**. The longest sentence is chosen to unfold LSTM model. We set the learning rate and learning rate decay at 0.007, 0.96 during training, respectively. Linear kernel is selected for SVM classifier. The SVM parameters are tuned using the validation set.

5 Result

Table 2 shows the result of the collated models and the proposed models on ROUGE-1 score, F1-score and CRate for the English dataset. The first four rows are the referenced result from related works. The cascaded model has the highest F1-score but it is hard to compare without compression rate. It is clear that the CRF-combined models and the bi-enc-dec models outperform other methods in both F1-score and CRate.

Table 2. Result on English dataset

Model	ROUGE-1	F1-Score	CRate
Seq-to-seq	-	74.31	0.45
Bi-LSTM [24]	-	75.05	0.47
t-attention [24]	-	76.98	0.45
Cascaded [12]	-	80.97	-
Bi-LSTM-CRF	82.11	78.59	0.38
Bi-enc-dec-CRF	80.21	78.21	0.40
Bi-enc-dec	80.07	77.12	0.40
Bi-enc-dec-SVM	82.02	77.64	0.42

Generally, the bi-enc-dec models are better at producing compression rate which are close to the golden rate although it fails to deliver advantageous information for used by CRF model. The result indicates the potential adaptability of the CRF-based models. Similar to the sequence tagging problems, the decision of deleting or keeping a token has its dependency on the decision of the previous and the following ones. CRF is built on the constraint between adjacent words so that CRF-based models can successfully capture the relationship

of close words, for example, the words in the same phrase. Whereas, the linear classifier and SVM deliver output labels independently therefore they cannot model that relation.

We also report the performance of our models on Vietnamese dataset in Table 3. Bi-enc-dec-based method outperforms other method in terms of ROUGE-score and F-score. Comparing to the Bi-LSTM based method, bi-enc-dec increases F-score while still generating shorter sentences.

Tables 4 and 5 show the sample outputs of the top compression systems. The words with the strikethrough indicate that they are unexpectedly deleted; the italic words are wrongly kept.

Table 3. Result on Vietnamese dataset.

Model	ROUGE-1	F1-score	CRate
CRF-MC [25]	80.97	-	0.79
Bi-LSTM-CRF	81.37	79.38	0.86
Bi-enc-dec-CRF	80.68	79.67	0.83
Bi-enc-dec	82.89	78.71	0.82
Bi-enc-dec-SVM	82.90	78.71	0.82

Table 4. Example original and compressed sentence on Vietnamese data set (1) Bi-LSTM-CRF, (2) Bi-enc-dec, (3) Bi-enc-dec-SVM.

Input	Liên minh châu Âu từng áp đặt lệnh trừng phạt với Jama 'a Jama 'a vào năm 2011 vì cho rằng ông có vai trò quan trọng trong việc đàn áp dân thường Syria .
	The European Union used to impose sanctions with Jama 'a Jama' in 2011 because of his important role in suppressing Syrian civilians .
Gold	Liên minh châu Âu từng áp đặt lệnh trừng phạt với Jama 'a Jama 'a vì cho rằng ông có vai trò quan trọng trong việc đàn áp dân thường Syria .
(1)	Liên minh châu Âu từng áp đặt lệnh trừng phạt với Jama 'a Jama 'a vào năm 2011 vì cho rằng ông có vai trò quan trọng trong việc đàn áp dân thường Syria .
(2)	Liên minh châu Âu từng áp đặt lệnh trừng phạt với Jama 'a Jama 'a vì cho rằng ông có vai trò quan trọng trong việc đàn áp dân thường Syria .
(3)	Liên minh châu Âu từng áp đặt lệnh trừng phạt với Jama 'a Jama 'a vì cho rằng ông có vai trò quan trọng trong việc đàn áp dân thường Syria .

Table 5. Example original and compressed sentence on English data set (1): Bi-LSTM-CRF, (2): Bi-enc-dec, (3): Bi-enc-dec-SVM

Input	Barack Obama has called for the release of an American pastor imprisoned in Iran, reports state
Gold	Barack Obama has called for the release of an American pastor imprisoned in Iran
(1)	Barack Obama has called for the release of an American pastor imprisoned in Iran
(2)	Barack Obama has called for the release of an American pastor imprisoned in Iran
(3)	Barack Obama has called for the release of an American pastor imprisoned in Iran
Input	Whole Foods is launching a new TV show called Dark Rye on the Pivot TV network
Gold	Whole Foods is launching a TV show
(1)	Whole Foods is launching a <i>new</i> TV show <i>called Dark Rye on the Pivot TV network</i>
(2)	Whole Foods is launching a <i>new</i> TV show <i>called Dark Rye</i>
(3)	Whole Foods is launching a <i>new</i> TV show <i>called Dark Rye</i>
Input	BJP on Sunday accused Delhi’s law minister Somnath Bharti of “racism” and wondered if the dharna planned by chief minister Arvind Kejriwal to demand suspension of some police officials is a ploy for “a way out of the government and get back to the streets”
Gold	BJP accused Somnath Bharti of racism
(1)	BJP accused <i>Delhi’s law minister</i> Somnath Bharti of racism
(2)	BJP accused <i>Delhi’s law minister</i> Somnath Bharti of racism
(3)	BJP accused <i>Delhi’s law minister</i> Somnath Bharti of racism
Input	Lehigh University President Alice P. Gast, the college’s 13th president in its 148-year history, will step down on July 31, according to a posting on the university website
Gold	Lehigh University President Alice P. Gast will step down
(1)	Lehigh University President Alice P. Gast will step down
(2)	Lehigh University President Alice P. Gast will step down <i>on July 31</i>
(3)	Lehigh University President Alice P. Gast will step down <i>on 31</i>
Input	A Zimbabwean pastor was jailed on Monday for 50 years for raping four female congregants and threatening to cast evil spells on them if they exposed him
Gold	A Zimbabwean pastor was jailed for raping
(1)	A Zimbabwean pastor was jailed <i>for 50 years</i> for raping <i>four female congregants</i>
(2)	A Zimbabwean pastor was jailed <i>for 50 years</i> for raping <i>four female</i>
(3)	A Zimbabwean pastor was jailed <i>for 50 years</i> for raping <i>four female</i>

6 Conclusion

We demonstrate various compression systems which request word embedding as the unique feature. We adopted the Bi-LSTM-CRF for this new problem and propose the combined model of Bi-enc-dec with either SVM or CRF. The models were evaluated on the public English and Vietnamese data sets.

The results from the two data sets illustrate that we can figure out the sentence compression problem by using various type of word embedding. The CRF-based models are good at modeling the adjacent relation while the combination of conventional classifier and bi-enc-dec show their efficiency in enhancing conventional deep neural network models. The bi-enc-dec model is helpful at producing expected compression rate. In the future, we would like to study the more challenging “abstractive text summarization” which encapsulates the content of the sentence in a shorter text sequence by introducing new words and phrases.

Acknowledgment. This work was supported by JSPS KAKENHI Grant number JP15K16048.

References

1. Cheng, J., Lapata, M.: Neural summarization by extracting sentences and words (2016)
2. Chopra, S., Auli, M., Rush, A.M., Harvard, S.: Abstractive sentence summarization with attentive recurrent neural networks. In: Proceedings of NAACL-HLT16, pp. 93–98 (2016)
3. Clarke, J., Lapata, M.: Global inference for sentence compression: an integer linear programming approach. *J. Artif. Intell. Res.* **31**, 399–429 (2008)
4. Cohn, T., Lapata, M.: Sentence compression beyond word deletion. In: COLING, pp. 137–144. Association for Computational Linguistics (2008)
5. Dyer, C., Ballesteros, M., Ling, W., Matthews, A., Smith, N.A.: Transition-based dependency parsing with stack long short-term memory (2015)
6. Filippova, K., Alfonseca, E., Colmenares, C.A., Kaiser, L., Vinyals, O.: Sentence compression by deletion with LSTMs. In: EMNLP, pp. 360–368 (2015)
7. Filippova, K., Altun, Y.: Overcoming the lack of parallel data in sentence compression. In: EMNLP, pp. 1481–1491. Citeseer (2013)
8. Filippova, K., Strube, M.: Dependency tree based sentence compression. In: INLG, pp. 25–32 (2008)
9. Hinton, G., Deng, L., Yu, D., Dahl, G.E., Mohamed, A.R., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T.N., et al.: Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. *IEEE Signal Process. Mag.* **29**, 82–97 (2012)
10. Hochreiter, S., Schmidhuber, J.: Long Short-Term Memory, vol. 9, pp. 1735–1780. MIT Press, Cambridge (1997)
11. Huang, Z., Xu, W., Yu, K.: Bidirectional LSTM-CRF models for sequence tagging. arXiv preprint [arXiv:1508.01991](https://arxiv.org/abs/1508.01991) (2015)
12. Klerke, S., Goldberg, Y., Søgaard, A.: Improving sentence compression by learning to predict gaze. In: Proceedings of NAACL-HLT, pp. 1528–1533 (2016)

13. Knight, K., Marcu, D.: Statistics-based summarization-step one: sentence compression 2000, pp. 703–710 (2000)
14. Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., Dyer, C.: Neural architectures for named entity recognition. arXiv preprint [arXiv:1603.01360](https://arxiv.org/abs/1603.01360) (2016)
15. Le, N.M., Horiguchi, S.: A new sentence reduction based on decision tree model. In: Proceedings of the 17th PACLIC, pp. 290–297 (2003)
16. Le Nguyen, M., Shimazu, A., Horiguchi, S., Ho, B.T., Fukushi, M.: Probabilistic sentence reduction using support vector machines. In: Proceedings of the 20th COLING, p. 743. Association for Computational Linguistics (2004)
17. Luong, M.T., Sutskever, I., Le, Q.V., Vinyals, O., Zaremba, W.: Addressing the rare word problem in neural machine translation (2014)
18. McDonald, R.T.: Discriminative sentence compression with soft syntactic evidence. In: EAACL (2006)
19. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: NIPS, pp. 3111–3119 (2013)
20. Pennington, J., Socher, R., Manning, C.D.: Glove: global vectors for word representation. In: EMNLP, vol. 14, pp. 1532–1543 (2014)
21. Qian, X., Liu, Y.: Fast joint compression and summarization via graph cuts. In: EMNLP, pp. 1492–1502 (2013)
22. Quirk, C., Brockett, C., Dolan, W.B.: Monolingual machine translation for paraphrase generation. In: EMNLP, pp. 142–149 (2004)
23. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: NIPS, pp. 3104–3112. MIT Press (2014)
24. Tran, N.T., Luong, V.T., Nguyen, N.L.T., Nghiem, M.Q.: Effective attention-based neural architectures for sentence compression with bidirectional long short-term memory. In: Proceedings of the 7th SoICT, pp. 123–130 (2016)
25. Tran, N.T., Ung, V.G., Luong, A.V., Nghiem, M.Q., Nguyen, N.L.T.: Improving Vietnamese sentence compression by segmenting meaning chunks. In: Knowledge and Systems Engineering (KSE), pp. 320–323 (2015)
26. Xian, Q., Yang, L.: Polynomial time joint structural inference for sentence compression, vol. 2, pp. 327–332. ACL (2014)

Text and Message Understanding



Myanmar Number Normalization for Text-to-Speech

Aye Mya Hlaing¹(✉), Win Pa Pa¹, and Ye Kyaw Thu²

¹ Natural Language Processing Lab, UCSY, Yangon, Myanmar
{ayemyahlaing, winpapa}@ucsy.edu.mm

² Artificial Intelligence Lab, Okayama Prefectural University, Okayama, Japan
ye@c.oka-pu.ac.jp

Abstract. Text Normalization is an essential module for Text-to-Speech (TTS) system as TTS systems need to work on real text. This paper describes Myanmar number normalization designed for Myanmar Text-to-Speech system. Semiotic classes for Myanmar language are identified by the study of Myanmar text corpus and Weighted Finite State Transducers (WFST) based Myanmar number normalization is implemented. Number suffixes and prefixes are also applied for token classification and finally, post-processing has been done for tokens that cannot be classified. This approach achieves average tag accuracy of 93.5% for classification phase and average Word Error Rate (WER) 0.95% for overall performance which is 5.65% lower than rule-based system. The results show that this approach can be used in Myanmar TTS system, and to our knowledge, this is the first published work of Myanmar number normalization system designed for Myanmar TTS system.

Keywords: Myanmar number normalization · Text normalization
Weighted finite state transducer · Myanmar text-to-speech · Myanmar

1 Introduction

Text-to-Speech (TTS) synthesis is a technique for generating intelligible, natural-sounding artificial speech for a given input text. Typical TTS systems have two main components, text analysis and waveform synthesis [1]. Text normalization, phonetic analysis, and prosodic analysis are needed to be done in text analysis components. Text normalization is the first and crucial phase of text analysis.

TTS systems need to work on real text and it contains non-standard words (NSWs), including numbers, dates, currency amounts, abbreviations, and acronyms. NSWs cannot be found in a dictionary or cannot get their pronunciations by “letter-to-sound” rules [2].

Myanmar text contains many NSWs with numbers. If normalization of these NSWs is not performed in the initial step of Myanmar TTS system, the quality of the system will be degraded. Therefore, normalization of NSWs with Myanmar numbers are more emphasized for this research.

Most text normalization still involves hand-constructed grammars because statistical techniques typically require at least some annotated data to train models, which is often not available for all of the ambiguities one would like to resolve [3].

Some information of rule-based text normalization for Myanmar language is described in [4]. Rule-based and lookup dictionary based methods are applied in text normalization of Croatian [5], Bengali [6] languages. Finite State Automata (FSA) and Maximum Entropy (ME) Classifier and Rules are used as the text normalization strategy of Mandarin [7] language. Application of decision tree and decision list are explored in Hindi text normalization [8]. Language model, supervised and unsupervised approaches are applied in Russian number names [9] and Vietnamese text normalization [10]. Weighted finite-state transducers (WFSTs) are applied for the Kestrel text normalization system, a component of the Google text-to-speech synthesis system [3]. Recurrent neural networks (RNN) are applied to learn the correct normalization function from a large corpus of written text aligned to its normalized spoken form. Although RNN produces good results on overall accuracy, it is prone to make the occasional errors [11].

For Myanmar language, statistical approaches cannot be used on number normalization because there is no annotated or parallel normalized corpus for Myanmar number normalization. Therefore, Myanmar number normalization is implemented by writing number normalization grammars that are compiled into libraries of WFST.

This paper describes the semiotic classes for Myanmar language and the development of number normalization for Myanmar text-to-speech system by implementing two phases: classification and verbalization phases. Word segmentation process is needed as the pre-processing step because Myanmar text lacks white space between words. Language specific grammars based on WFST are applied for this system.

2 Defining Semiotic Classes for Myanmar Language

The first part of number normalization is a classification of NSWs and the second part is a verbalization of detected NSWs into their standard words. For NSWs classification, the semiotic classes for Myanmar language are needed to identify. Based on the investigation of Myanmar sentences from Asian Language Treebank (ALT) parallel corpus¹[12] and some Web data (3,150 sentences), we identified a set of semiotic classes for Myanmar Language. ALT corpus is the parallel corpus for seven languages in the news domain and comprises 20,000 sentences. The result semiotic classes are shown in Table 1. NSWs (Myanmar digits) are shown in bold style in tables and contents of the paper.

3 Defining Rules for Number Normalization

For detection of the semiotic classes defined in Table 1, 60 rules are defined for simple rule-based number normalization system. Rule-based number normaliza-

¹ <http://www2.nict.go.jp/astrec-att/member/mutiyama/ALT/index.html>.

Table 1. Semiotic classes for myanmar

Semiotic Class	Description	Example:
DATE	date	၁၂-၂-၂၀၁၇, ၂၀၁၇ ဖေဖော်ဝါရီ ၁၂, etc.
TIME	time	၁၀:၂၀:၂၅, ၁၀ နာရီ ၂၀ မိနစ်, ဂျီအမ်တီ ၀၈၃၀, etc.
CURRENCY	currency amount	၁၂၃,၄၅၀ ကျပ်, \$၃,၀၀၀, etc.
NUMBER	cardinal, decimal	၁၀ ယောက်, ၁၂၅.၄၅, ၆၀% , etc.
DIGIT	digit by digit	+၉၅-၉ ၄၃၀၅၄၄၈၅, အမှတ် ၄၄၇, etc.
RANGE	range	၇၀ - ၈၀ ဒီဂရီဖာရင်ဟိုက်, ဒေါ်လာ ၁၀၀ နှင့် ၁၅၀ ကြား, etc.
SCORE	score	၂-၃ ဝိုး, ၂:၃ ဝိုး, etc.
DIMENSION	dimension	ပေ ၄၀ x ၆၀, etc.
NRC	national identification number	၅/မရန(နိုင်) ၁၂၃၄၅၆, etc.

tion is implemented by regular expression (RE) in Perl programming language. Look-up dictionary is used for expansion of measurement units and currency unit symbols. An example RE for detecting date range in DATE semiotic class is as follows:

```
if($inputdata =~ m/(\s+[0-9]{4}\s*)-(\s*[0-9]{4}\s*) ($strTwoDateSuf)/g)
```

RE for detecting currency with currency suffix is written as follows:

```
if($inputdata =~ m/([0-9]{1,3},([0-9]{3},)*)[0-9]{3}) (\.? [0-9]*\s*) ($strCurrencySuf)/g)
```

This rule-based number normalization system is used as a baseline in this paper.

4 Weighted Finite State Transducer for Number Normalization

The main point of this paper is language-specific grammar that is compiled to WFST. A WFST consists of a set of states and transitions between states. Each

transition is labeled with an input symbol from an input alphabet; an output symbol from an output alphabet; an origin state; a destination state; and a weight [13]. Finite State Transducer (FST) can represent certain sets and binary relations over the string. A Deterministic Finite Automaton recognizes a regular language and FST recognizes a regular relation. FSTs have been used to model the morphology, phonology and other text-analysis operations such as numeral expansion. WFSTs have been used in Text analysis model for Text-to-Speech (TTS) of the multilingual Bell Labs TTS system [14].

OpenGrm Thrax Grammar Compiler² [13] is used for the development of Myanmar number normalization system. The Thrax grammar compiler compiles grammars that consist of regular expressions, and context-dependent rewrite rules, into FST archives of weighted finite state transducers. It uses OpenFst library to provide an efficient encoding of grammars and general algorithms.

5 Myanmar Number Normalization

Myanmar number normalization is achieved by implementing two phases: classification and verbalization phases. Both phases are accomplished by defining grammars that are compiled to WFST. Semiotic classes defined in Sect. 2 is used for classification. Number prefixes and suffixes are also used as the clues for identifying the semiotic class of the current word. Word segmentation process is needed as the pre-processing step of this system. Detailed processes for the whole system are discussed in the following sections.

5.1 Grammars for Classification

Grammars that are compiled to the libraries of WFST are applied for classification phase. Before applying these grammars, some pre-processing rules are written for combining tokens. Some digit sequences need to be combined for classifying their semiotic classes. For example, in a Myanmar sentence,

“မီတာ ၁၀၀ ၊ ၂၀၀ ၊ ၃၀၀ စီ ဝေးပါတယ်။”

(There are 100 m, 200 m and 300 m far from here.)

In this case, “မီတာ” (meter) is the clue for identifying the whole digit sequence as the NUMBER class and it is located before the first digit. The other digits need to be related that clue. Spaces between these digit sequences are removed as the pre-processing step and it becomes “မီတာ ၁၀၀၂၀၀၃၀၀ စီ ဝေးပါတယ်။”. Therefore, the whole digit sequence can be identified as the NUMBER class.

For DATE semiotic class, more than one rule is needed to implement because many patterns of DATE are commonly written in Myanmar. Table 2 shows some examples of Myanmar date.

There are two types of clues in classification for CURRENCY semiotic class. Currency symbol can be used as the prefix of the number or currency text as the prefix or suffix of the number. As an example, “\$ ၅၀”, “ဒေါ်လာ ၅၀” and “၅၀ဒေါ်လာ” has the same meaning in English as “\$50”.

² <http://www.openfst.org/twiki/bin/view/GRM/Thrax>.

Table 2. Myanmar dates

Myanmar	English
ဇန်နဝါရီ ၄၊ ၂၀၁၇	January 4 2017
၂၀၁၇ ၊ ဇန်နဝါရီလ ၄	2017 January 4
၂၀၁၇ ခုနှစ် ၊ ဇန်နဝါရီလ ၄	2017 January 4
၂၀၁၇ ၊ ၄ လပိုင်း	4/2017
ဇန်နဝါရီလ ၄	January 4
၄.၁.၂၀၁၇	4.1.2017
၄-၁-၂၀၁၇	4-1-2017
၄/၁/၂၀၁၇	4/1/2017
၁၃၇၈ ခုနှစ် ဝါဆို လဆန်း ၁၀ ရက်	10, waxing Warso, 1378
၁၃၇၈ ခုနှစ်၊ တန်ဆောင်မုန်း လပြည့်ကျော် ၁၀ ရက်	10, waning Tazaungmone, 1378
၂၀၁၆-၂၀၁၇ ခုနှစ်	2016-2017
၂၀၁၆-၂၀၁၇ ပညာသင်နှစ်	2016-2017 academic year

In this paper, cardinal, decimal and measure are defined in the member of NUMBER semiotic class. In Myanmar, ordinal numbers are usually written by using ordinary number or text.

Examples are “၂ ကြိမ်မြောက်” (second) and “ပထမအကြိမ်” (first).

The first example is an ordinal number using ordinary number that means “second” in English and the second one is using text that means “first” in English. Therefore, it is not needed to be identified as one type of semiotic class. Measure unit symbols, 96 number prefixes, and 275 number suffixes are used for classification of NUMBER semiotic class. These prefixes and suffixes are extracted by collecting manually from Myanmar text corpus. Measure unit symbols with their expansions like “% (ရာခိုင်နှုန်း)”, “°F (ဒီဂရီဖာရင်ဟိုက်)”, “ft (ပေ)” are applied by using StringFile function.

Finally, grammars for all defined semiotic classes are compiled to WFSTs and then these WFSTs are exported and used in Classification phase. Weights are assigned for defining different priorities of the classification. Example input string and output string of classification phase are as follows:

Input String: မြန်မာ နှစ်ဆန်းတစ်ရက်နေ့သည် ၂၀၁၇ ဧပြီလ ၁၇ ရက်နေ့ ဖြစ်ပါသည်။
(Myanmar New Year’s day falls on 17th April, 2017.)

Output String: မြန်မာ နှစ်ဆန်းတစ်ရက်နေ့သည် <DATE>year: ၂၀၁၇ month: ဧပြီလ day: ၁၇</DATE> ရက်နေ့ ဖြစ်ပါသည် ။

5.2 Grammars for Verbalization

Verbalization is the expansion of tokens into standard words and its expansion mainly depends on its semiotic class. Verbalization of semiotic classes that comes out from classification phase is also done by using WFST. In Myanmar language, some symbols have to be neglected in verbalization. For example, in DIMENSION semiotic class,

“ပေ ၄၀ × ၆၀” (40 ft × 60 ft) can be expanded into

“ပေ လေး ဆယ် ခြောက် ဆယ်”.

In this case, “×” symbol must be neglected in verbalization.

A score, “၃-၁ ဝိုး” (3:1) can be expanded into

“သုံး ဝိုး တစ် ဝိုး”

In the above SCORE case, “-” symbol needs to be replaced by suffix “ဝိုး” . In Myanmar, “-” is usually used in SCORE case.

Myanmar people sometimes omit “တစ်” (one) in pronunciation of Myanmar digit sequence like “တစ်ဆယ့်” (ten) and “တစ်ထောင်” (one thousand). Examples are:

- | | |
|--------------------------------------|--------------------------|
| ၁၁:၀၀ နာရီ (11:00 a.m) | - ဆယ့် တစ် နာရီ |
| ၁၉၁၅ (1915) | - ထောင် ကိုး ရာ ဆယ့် ငါး |
| ၁၅၀၀ ကျပ် (1500 Kyats ⁵) | - ထောင် ငါး ရာ ကျပ် |

Therefore, some rules for fixing these particular cases are written in verbalization grammars.

5.3 Myanmar Number Names Expansion

All the grammars except digit by digit expansion depend on the expansion of Myanmar number names. The number name grammars depend on the factorization of digit string into sum of products of powers of ten. These factorizations can be done by using Thrax Factorizer grammar. Number verbalization of these factorized strings into number names depends on the language. Most Western languages have no terms for 10^4 , they say ten thousand. The factorization becomes $1 \times 10^1 \times 10^3$. In Myanmar language, we have a term “ထောင်” (ten thousand) for 10^4 and the factorization is 1×10^4 . Although there are terms “သန်း” (million) for 10^6 and “ကိုဠ” (ten million) for 10^7 in Myanmar language, it is not commonly used in pronunciation of CURRENCY class. Myanmar people usually say “၁၀ သိန်း” (ten lakh) for 10^6 and “သိန်း ၁၀၀” (hundred lakh) for 10^7 . Therefore, factorization defines $1 \times 10^1 \times 10^5$ for the first case and $1 \times 10^2 \times 10^5$ for the second case and Myanmar number names grammar has to handle these

factorization into appropriate Myanmar standard words. As an example,

“၁,၂၃၄,၅၆၇,၈၉၀” (1,234,567,890) will be expanded into standard words as

“တစ် သောင်း နှစ် ထောင် သုံး ရာ လေး ဆယ့် ငါး သိန်း ခြောက် သောင်း ခုနစ် ထောင် ရှစ် ရာ ကိုး ဆယ်”.

(twelve thousand three hundred forty-five lakh sixty-seven thousand eight hundred and ninety).

5.4 Finalization

After verbalization phase, the system missed identifying the semiotic classes of some digit sequences and verbalizing into their standard words because they have no clue for defining semiotic classes. In finalization, post-processing have to be done for fixing these cases. Some assumptions are made to decide whether current token should be classified as cardinal number or as digit by digit. If there is a comma or dot in digit sequence or it has one non-zero digit followed by many zero digits, it will be expanded into cardinal number names. If not, it will be checked how many digits the sequence has. If it has three or more digits, it will be pronounced as digit by digit and if it has less than three digits, pronounced as the cardinal number. Myanmar people usually pronounce as that way. For example, Bus no. “၃၉” (39) is usually pronounced as “သုံး ဆယ့် ကိုး” (thirty-nine) and Bus no. “၁၂၄” (124) as “တစ် နှစ် လေး” (one two four).

6 Sample Grammar Fragments

In classification phase, as we mentioned in Sect. 5.1, some clues are used for classifying semiotic class. However, if there is a number sequence of same semiotic class, it is common that clue is located at the start or end of the sequence of numbers separated by Myanmar punctuation “၊”. As an example, a sequence of currency number is written as

“၂၀,၀၀၀ ၊ ၁၅,၀၀၀ ၊ ၁၀,၀၀၀ ကျပ် (20,000, 15,000, 10,000 kyats) instead of writing “၂၀,၀၀၀ ကျပ်၊ ၁၅,၀၀၀ ကျပ်၊ ၁၀,၀၀၀ ကျပ်” (20,000 kyats, 15,000 kyats, 10,000 kyats).

Example grammar for detecting sequence of currency is as follows:

For example: ၂၀,၀၀၀ ၊ ၁၅,၀၀၀ ၊ ၁၀,၀၀၀ ကျပ် => <CURRENCY>integer: ၂၀,၀၀၀ ၊ integer: ၁၅,၀၀၀ ၊ integer: ၁၀,၀၀၀ currencySuffix: ကျပ် </CURRENCY>

```

currency_sequence =
  u.Ins["<CURRENCY>"]
  u.Ins[" integer: "]
  validnum
  (u.Del["."]
  u.Ins[" fractional_part: "]
  d+)? "1"
  (u.Ins[" integer: "]
  validnum
  (u.Del["."]
  u.Ins[" fractional_part: "]
  d+)? "1" )*
  u.Ins[" integer: "]
  validnum
  (u.Del["."]
  u.Ins[" fractional_part: "]
  d+)?
  u.Del[" **"]
  u.Ins[" currencySuffix: "]
  currency_suffix
  u.Ins["</CURRENCY>"]

```

In verbalization, Myanmar number name grammar is used as the basic and some rewrite rules are written to fix some particular cases. For example, omitting the pronunciation of “တစ်” before “ဆယ့်” and “ထောင်” is written by using content-dependent rewrite rule and composition is applied in this case. Example code fragment is as follows:

For example: ၁၁-၃-၁၉၁၇ (11-3-1917) => ဆယ့် တစ် ရက် သုံး လ ထောင် ကိုး ရာ ဆယ့် ခုနှစ်

```

day = n.MYANMAR_NUMBER_NAME;
month_num = n.MYANMAR_NUMBER_NAME;
year = n.MYANMAR_NUMBER_NAME;
date_num =
  u.Del["<DATE>"]
  u.Del[" day: "]
  day
  u.Ins[" ရက် " ]
  u.Del[" month: "]
  month_num
  u.Ins[" လ "]
  u.Del[" year: "]
  year
  u.Del["</DATE>"];
remove_tit = CDRewrite["တစ်" : "" , "", (" ဆယ့် ") | (" ထောင် "), sigma_star];
export DATE_VERBALIZE = Optimize[date_num @ remove_tit];

```

7 Test Data Preparation

Two test data TestData-1 and TestData-2 are prepared for evaluating current WFST-based Myanmar number normalization. TestData-1 has 1,000 sentences which are randomly selected from ALT corpus that needs to be normalized. For TestData-2, some Myanmar sentences from the Web that contains 947 sentences with Myanmar digits are selected. For these two test data, parallel tagged corpus is prepared for evaluating classification and parallel normalized corpus for evaluating the overall performance.

8 Experimental Results

Experiments are designed to test the performance of classification and verbalization of Myanmar number normalization.

8.1 Classification

The following formula is used for evaluating the performance of classification.

$$\text{tag accuracy}(\%) = \frac{\text{number of particular tags in test data}}{\text{number of particular tags in reference data}} \quad (1)$$

In the formula, the numerator is the number of output tags from the system and the denominator is the number of expected tags.

Table 3 shows the number of particular tags in test data and reference data and the percentage of tag accuracy. The overall tag accuracy is **94.3%** on TestData-1 and **92.6%** on TestData-2.

Table 3. Classification accuracy of two test data

Semiotic class	TestData-1	TestData-2
NUMBER	755/810 (93.2%)	725/789 (91.9%)
DATE	396/404 (98%)	422/442 (95.5%)
TIME	78/80 (97.5%)	68/68 (100%)
CURRENCY	71/82 (86.6%)	98/118 (83.1%)
DIGIT	5/5 (100%)	3/3 (100%)
RANGE	2/2 (100%)	6/8 (75%)
SCORE	5/8 (62.5%)	N/A
DIMENSION	N/A	2/2 (100%)
Overall Accuracy	1312/1391 (94.3%)	1324/1430 (92.6%)

8.2 Verbalization

Verbalization results are reported in terms of word error rate (WER), a standard for evaluation of Automatic Speech Recognition system. Simple rule-based number normalizer developed by Perl programming language is used as the baseline in this paper. In Table 4, current number normalization system based on WFST achieves WER **0.5%** for TestData-1 and **1.4%** for TestData-2, and which is **5.0%** and **6.3%** lower than the baseline system respectively.

Table 4. Overall performance

	TestData-1 (WER%)	TestData-2 (WER%)
Baseline	5.5%	7.7%
WFST-based	0.5%	1.4%

9 Error Analysis and Discussion

Some errors are found in classification phase in our experiments because of two factors. The first factor is that there is no clue before and after the digit sequence.

In this example, ၇၀ ကနေ ၃၁၀ ကီလိုမီတာ (from 70 to 310 km) should be classified as

<NUMBER> ၇၀ </NUMBER> ကနေ <NUMBER> integer: ၃၀၀ numberSuffix: ကီလိုမီတာ </NUMBER>.

However, the classifier cannot classify which class should be occupied ၇၀ because it has no clue.

The second factor is that there are some common prefixes or suffixes on both CURRENCY and NUMBER classes. For example, in this Myanmar sentence,

ကမ္ဘာ့လူဦးရေက လတိုင်း ၆ သန်း လောက်တိုးပွားနေပါတယ်။ (The world’s population has been increasing about 6 million per month.),

although ၆ သန်း (6 million) should be classified as the NUMBER class, its result class is CURRENCY because သန်း (million) is also the prefix of CURRENCY class.

As the same way, CURRENCY class is sometimes misclassified as NUMBER class. For example, in the sentence,

အမေရိကန်ဒေါ်လာ သန်းပေါင်း ၃၈၀ ခန့် ရင်းနှီးမြှုပ်နှံမည်။ (It will invest 380 million dollars approximately.), the prefix of the digit sequence သန်းပေါင်း is the prefix of CURRENCY class and the suffix ခန့် is the suffix used in NUMBER class. Therefore, a classifier based on FST chose the shorter path ၃၈၀ ခန့် as the NUMBER class.

Although there are some classification errors, verbalization results of some misclassification have also correct pronunciation because of almost same pronunciation for CURRENCY and NUMBER class in Myanmar language and post-processing of our number normalization. Therefore, low WER on overall accuracy is achieved in our experiments.

10 Conclusion and Future Work

Number normalization is a very important module in Text-to-Speech system. In this paper, semiotic classes for Myanmar language are identified and WFST-based Myanmar number normalization designed for Myanmar TTS system is implemented for the first time. Experimental results show that this WFST-based approach can get acceptable results for the performance of Myanmar number normalization and this can be used practically by integrating with Myanmar TTS system.

For solving misclassification and missing tag cases in classification phase, this phase will be replaced by applying sequence-to-sequence modeling. A tagged corpora for various types of data resources will be built for that modeling.

Acknowledgements. This work is partly supported by the ASEAN IVO project “Open Collaboration for Developing and Using Asian Language Treebank”.

References

1. Taylor, P.: Text-to-Speech Synthesis. Cambridge University Press, Cambridge (2009)
2. Sproat, R., Black, A.W., Chen, S., Kumar, S., Ostendorf, M., Richards, C.: Normalization of non-standard words. *Comput. Speech Lang.* **15**(3), 287–333 (2001)
3. Ebden, P., Sproat, R.: The kestrel TTS text normalization system. *Nat. Lang. Eng.* **21**(03), 333–353 (2015)
4. Thu, Y.K., Pa, W.P., Ni, J., Shiga, Y., Finch, A., Hori, C., Kawai, H., Sumita, E.: Hmm based myanmar text to speech system. In: Sixteenth Annual Conference of the International Speech Communication Association (2015)
5. Beliga, S., Martinčić-Ipšić, S.: Text normalization for croatian speech synthesis. In: MIPRO, 2011 Proceedings of the 34th International Convention, pp. 1664–1669. IEEE (2011)
6. Alam, F., Habib, S., Khan, M.: Text normalization system for bangla. Technical report, BRAC University (2008)
7. Zhou, T., Dong, Y., Huang, D., Liu, W., Wang, H.: A three-stage text normalization strategy for mandarin text-to-speech systems. In: 6th International Symposium on Chinese Spoken Language Processing, ISCSLP 2008, pp. 1–4. IEEE (2008)
8. Panchapagesan, K., Talukdar, P.P., Krishna, N.S., Bali, K., Ramakrishnan, A.: Hindi text normalization. In: Fifth International Conference on Knowledge Based Computer Systems (KBCS), pp. 19–22. Citeseer (2004)
9. Sproat, R.: Lightly supervised learning of text normalization: Russian number names. In: 2010 IEEE Spoken Language Technology Workshop (SLT), pp. 436–441. IEEE (2010)
10. Nguyen, T.T.T., Pham, T.T., Tran, D.D.: A method for vietnamese text normalization to improve the quality of speech synthesis. In: Proceedings of the 2010 Symposium on Information and Communication Technology, pp. 78–85. ACM (2010)
11. Sproat, R., Jaitly, N.: RNN approaches to text normalization: a challenge. arXiv preprint [arXiv:1611.00068](https://arxiv.org/abs/1611.00068) (2016)

12. Riza, H., Purwoadi, M., Gunarso, Uliniansyah, T., et al.: Introduction of the asian language treebank. *Oriental COCOSDA* (2016)
13. Roark, B., Sproat, R., Allauzen, C., Riley, M., Sorensen, J., Tai, T.: The opengrm open-source finite-state grammar software libraries. In: *Proceedings of the ACL 2012 System Demonstrations*, pp. 61–66. Association for Computational Linguistics (2012)
14. Sproat, R.: Multilingual text analysis for text-to-speech synthesis. In: *Proceedings of the Fourth International Conference on Spoken Language, ICSLP 1996*, vol. 3, pp. 1365–1368. IEEE (1996)



Expect the Unexpected: Harnessing Sentence Completion for Sarcasm Detection

Aditya Joshi^{1,2,3}(✉), Samarth Agrawal², Pushpak Bhattacharyya²,
and Mark J. Carman³

¹ IITB-Monash Research Academy, Mumbai, India
adityaj@cse.iitb.ac.in

² Indian Institute of Technology Bombay, Mumbai, India
{samartha,pb}@cse.iitb.ac.in

³ Monash University, Melbourne, Australia
mark.carman@monash.edu

Abstract. The trigram ‘*I love being*’ is expected to be followed by positive words such as ‘*happy*’. In a sarcastic sentence, however, the word ‘*ignored*’ may be observed. The expected and the observed words are, thus, incongruous. We model sarcasm detection as the task of detecting incongruity between an observed and an expected word. In order to obtain the expected word, we use Context2Vec, a sentence completion library based on Bidirectional LSTM. However, since the exact word where such an incongruity occurs may not be known in advance, we present two approaches: an All-words approach (which consults sentence completion for every content word) and an Incongruous words-only approach (which consults sentence completion for the 50% most incongruous content words). The approaches outperform reported values for tweets but not for discussion forum posts. This is likely to be because of redundant consultation of sentence completion for discussion forum posts. Therefore, we consider an oracle case where the exact incongruous word is manually labeled in a corpus reported in past work. In this case, the performance is higher than the all-words approach. This sets up the promise for using sentence completion for sarcasm detection.

Keywords: Sarcasm detection · Sentence completion
Sentiment analysis · LSTM

1 Introduction

Sarcasm is defined as “*the use of irony to mock or convey contempt*”¹. For example, the sentence ‘*I love being ignored*’ is sarcastic. Automatic sarcasm detection is the task of predicting whether or not a given text contains sarcasm. Several statistical approaches have been proposed for sarcasm detection [1–3].

¹ Source: Oxford Dictionary.

In addition, rule-based approaches based on evidences of sarcasm have also done well [4–6]. This paper presents another rule-based technique. Our technique is novel in its application of sentence completion for sarcasm detection.

As an introduction to the technique, consider the sarcastic sentence ‘*I love being ignored*’. A likely word to follow the trigram ‘*I love being*’ would be a positive sentiment word such as ‘*happy*’. However, in the sarcastic sentence, the word ‘*ignored*’ occurs. The word ‘*ignored*’ in the sarcastic sentence is semantically distant from an expected word such as ‘*happy*’. This (dis)similarity can be used as an indicator of incongruity which is central to sarcasm, as per linguistic studies [7,8]. In order to obtain the expected word at a given position, we harness automatic sentence completion. Sentence completion predicts the most likely word at a given position in a sentence [9]. For our experiments, we use context2vec, a sentence completion toolkit [10]. Thus, our paper deals with the question:

Because incongruity in sarcasm is a phenomenon where the unexpected is observed, can sarcasm be detected using sentence completion?

A key assumption here is that a sentence completion toolkit trained on a large, general-purpose corpus follows the language model for non-sarcastic text. The assumption is reasonable because the sentence completion model is likely to have learned the language model for non-sarcastic text since sarcasm is an infrequent phenomenon.

It must be noted that the exact observed word where the incongruity occurs (‘*ignored*’ in the example above) is not known in advance. Hence, a sentence contains multiple candidate words of incongruity, out of which the incongruity is observed in case of specific word(s). We refer to these words as the ‘*incongruous word(s)*’. Therefore, our approaches vary in terms of the candidate incongruous words that are considered.

The novelty of this paper is as follows:

1. Using sentence completion for sarcasm detection
2. Experimentation with short text (where candidate incongruous words are a small set of words), long text (where candidate incongruous words are a large set of words), and an oracle case (where the exact incongruous word is known)

The rest of the paper is organized as follows. Section 2 describes the related work. Section 3 presents the motivation behind using sentence completion. Section 4 presents two approaches: an all-words approach, and an incongruous words-only approach. As stated earlier, the two approaches differ in terms of candidate incongruous words. Section 5 gives the experiment setup while Sect. 6 presents the results. We discuss an oracle case scenario in Sect. 7 to validate the strength of our hypothesis. Finally, we analyze the errors made by our system in Sect. 8 and conclude the paper in Sect. 9.

2 Related Work

The majority of the past work in statistical sarcasm detection uses sarcasm-specific features such as punctuations, emoticons or sarcasm-indicating n-grams

[1–3, 11, 12]. For example, [1] present a semi-supervised algorithm that first extracts sarcasm-indicating n-grams and then use them as features for a classifier. [11] use features based on number of sentiment flips, positive/negative subsequences, in addition to such n-grams. [3] include features such as audience information, twitter familiarity, etc.

Recent work in sarcasm detection employs features that capture contextual information such as an author’s background, conversational context, etc. [13–17]. Formulations beyond classifiers have also been considered. For example, [17] use sequence labeling algorithms to predict sarcasm in individual utterances in a dialogue. On the other hand, [16] use them to predict sarcasm of the last utterance in a dialogue with automatic labels in the rest of the sequence. However, in our case, we do not use any contextual information from the author or the conversation. This means that a hyperbolic sentence such as ‘*X is the best President ever!*’ (where the sarcasm cannot be understood based on the text alone) is beyond the scope of our approach.

In addition to the above, several rule-based techniques based on intuitive indicators of sarcasm have been reported. [6] predict a tweet as sarcastic if sentiment in the text of the tweet contradicts with the sentiment of a hashtag in the tweet. [4] predict a tweet as sarcastic if sentiment of the tweet does not match with sentiment of past tweets by the author of the tweet towards the entities in the tweet. Similarly, [5] use a set of nine rules to predict if a given simile (for example, ‘*as exciting as a funeral*’) is sarcastic. [12] capture sarcasm as a combination of positive verbs followed by negative situation phrases. Our approach is rule-based as well.

Our work is the first to employ sentence completion for the purpose of sarcasm detection. Sentence completion approaches based on word embeddings have been reported [18, 19]. However, they are only for sentence completion and not for sarcasm detection. They restrict themselves to completing sentences. We propose and validate the hypothesis that a ‘language model incongruity’ as experienced by a sentence completion module can be useful for sarcasm detection. We use context2vec [10] as the sentence completion library. The distinction between these sentence completion approaches is beyond the scope of this paper because the focus is to use one of them for sarcasm detection and demonstrate that it works.

3 Motivation

As stated in the previous section, in the sarcastic example ‘*I love being ignored*’, the word ‘*ignored*’ is observed at a position where positive sentiment words would be expected. Hence, the word ‘*ignored*’ is the exact incongruous word. Specifically, if context2vec [10] were consulted to complete the sentence ‘*I love being []*’ where [] indicates the position for which the most likely word is to be computed, the word ‘*happy*’ is returned. Word2vec similarity between ‘*happy*’ and ‘*ignored*’ is 0.0204, for certain pre-trained word2vec embeddings. This low value of similarity between the expected and observed words can be harnessed

as an indicator for sarcasm. In the rest of the paper, we refer to the word present at a given position as the ‘**observed word**’ (*‘ignored’* in the example above) where the most likely word at the position as returned by sentence completion is the ‘**expected word**’ (*‘happy’* in the example above).

However, a caveat lies in determination of the candidate incongruous words for which sentence completion will be consulted. For example, the sentence ‘*I could not make it big in Hollywood because my writing was not bad enough*’ is sarcastic because of the incongruous word ‘*bad*’ which is at the penultimate position in the sentence. In the absence of the knowledge of this exact incongruous word, it is obvious that an algorithm must iterate over a set of candidate incongruous words. Hence, we present two approaches: one which iterates over all words and another which restricts to a subset of words. The first approach is called the all-words approach, while the second is incongruous words-only approach. These approaches are described in detail in the next section. The ‘*oracle case*’ for our algorithm is a situation where the incongruous word is exactly known. We validate that our algorithm holds benefit even for the oracle case, in Sect. 7.

4 Approach

We present two approaches that use sentence completion for sarcasm detection: (a) an “all-words” approach, and (b) “incongruous words-only” approach. As stated earlier, in the absence of the knowledge about the exact position of incongruity, our technique must iterate over multiple candidate positions. For both the approaches, the following holds:

Input: A text of length l

Output: Sarcastic/non-sarcastic

Parameters:

- Similarity measure $sim(w_i, w_k)$ returning the similarity between words w_i and w_k
- Threshold T (a real value between minimum and maximum value of $sim(w_i, w_k)$)

4.1 All-Words Approach

As the name suggests, this approach considers all content words² as candidate incongruous words. This approach is as follows:

² Content words are words that are not function words. We ignore function words in a sentence.

```

min ← ∞
for p = 1 to l do:
    % compute expected word:
    ep ← context2vec(w1, ..., wp-1, [], wp+1, ..., wl)
    % check similarity to observed word:
    if sim(ep, wp) < min then min ← sim(ep, wp)
if min < T then predict sarcastic
    
```

Thus, for the sentence ‘A woman needs a man like a fish needs a bicycle’³ containing five content words (out of which ‘needs’ occurs twice), the sentence completion library will be consulted as follows:

1. A [] needs a man like a fish needs a bicycle.
2. A woman [] a man like a fish needs a bicycle.
3. A woman needs a [] like a fish needs a bicycle.
4. A woman needs a man like a [] needs a bicycle.
5. A woman needs a man like a fish [] a bicycle.
6. A woman needs a man like a fish needs a [].

4.2 Incongruous Words-Only Approach

A key shortcoming of the previous approach is that it may use similarity values for words which are not incongruous, since it makes six calls in case of the example given. For example, the first part of the sentence does not contain a language model incongruity and hence, the calls are redundant. Our second approach, the Incongruous words-only approach, reduces the set of words to be checked by sentence completion to half, thereby eliminating redundant comparisons as shown in the previous subsection. Incongruous words-only approach is as follows:

```

for p = 1 to l do:
    % compute average similarity to words:
    s̄p ← 1/(l-1) ∑i≠p sim(wi, wp)
    % choose positions with lowest averages:
    Incongruous ← {i : s̄i ≤ median(s̄1, ..., s̄l)}
min ← ∞
for p ∈ Incongruous do:
    % compute expected word:
    ep ← context2vec(w1, ..., wp-1, [], wp+1, ..., wl)
    % check similarity to observed word:
    if sim(ep, wp) < min then min ← sim(ep, wp)
if min < T then predict sarcastic
    
```

As seen above, we first select the required subset of words in the sentence. Beyond that, the approach is the same as the all-words approach. As a result, for the sentence ‘A woman needs a man like a fish needs a bicycle’, ‘fish’, ‘needs’

³ <http://www.phrases.org.uk/meanings/414150.html>.

and ‘*bicycle*’ are returned as most incongruous Incongruous words-only. Hence, the sentence completion is now consulted for the following input strings:

1. A woman [] a man like a fish needs a bicycle.
2. A woman needs a man like a [] needs a bicycle.
3. A woman needs a man like a fish [] a bicycle.
4. A woman needs a man like a fish needs a [].

We hope that this reduction in the set of candidate strings increases the chances of the algorithm detecting the incongruous word and hence, the sarcasm. We observe an interesting trend in short versus long text in terms of this reduction, as will be discussed in the forthcoming sections.

5 Experiment Setup

Since our approaches are contingent on the set of candidate phrases being considered, we consider two scenarios: short text where the set of words where incongruity has likely occurred is small, and long text where the set is large. Therefore, the two datasets used for the evaluation of our approaches are: (a) Tweets by [12] (2278 total, 506 sarcastic, manually annotated), and (b) Discussion forum posts by [20] (752 sarcastic, 752 non-sarcastic, manually annotated). We ignore function words when we iterate over word positions. They are not removed because such removal would disrupt the sentence, which is undesirable since we use sentence completion. We use a list of function words available online⁴.

For both approaches, we repeat the experiments over a range of threshold values, and report the best results (and the corresponding threshold values). As similarity measures, we use (a) **word2vec similarity** computed using pre-trained embeddings given by the Word2Vec tool. These embeddings were learned on the Google News corpus⁵, (b) **WordNet similarity** from WordNet::similarity by [21] (specifically, Wu-Palmer Similarity). The word2vec similarity in Incongruous words-only approach is computed in the same manner as word2vec similarity above. Since word2vec similarity may not be low for antonyms, we set the similarity measure for antonyms as 0. As stated earlier, for sentence completion, we use context2vec by [10]. It is a sentence completion toolkit that uses Bidirectional LSTM to predict a missing word, given a sentence. We use the top word returned by context2vec, as per the model trained on UkWac corpus⁶.

We report our evaluation for two configurations:

1. *Overall Performance*: In the first case, we run the algorithm for a range of threshold values and report results for the complete dataset.

⁴ <http://www.ranks.nl/stopwords>.

⁵ <https://code.google.com/archive/p/Word2Vec/>.

⁶ <http://u.cs.biu.ac.il/~nlp/resources/downloads/context2vec/>.

2. *Two-fold cross-validation:* Our algorithm is dependent on the value of the threshold. Hence, we divide the dataset into two splits and repeat the experiments in two runs: estimate the optimal threshold on a split, and report results for the other, and vice versa.

6 Results

In this section, we present an evaluation of our approaches, on the two datasets: the first consisting of short text (tweets), and the second consisting of long text (discussion forum posts). We first show the results for the complete dataset with optimal values of the threshold. We then repeat our experiments where the threshold is determined using a train-test split. These two configurations (overall performance and two-fold cross-validation) collectively validate the benefit of our approach.

6.1 Overall Performance

Table 1 shows the performance for tweets. Figure 1 shows how optimal thresholds are determined. When word2vec similarity is used for the all-words approach, an F-score of 54.48% is obtained. We outperform two past works [11, 12] which have reported their values on the same dataset. The best F-score of 80.24% is obtained when WordNet is used as the similarity measure in our Incongruous words-only approach. We observe that the Incongruous words-only approach performs significantly better than the all-words approach. In case of word2vec similarity, the F-score increases by around 18%, and by around 9% in case of WordNet similarity. Also, the optimal threshold values are lower for the all-words approach as compared to the Incongruous words-only approach.

Table 2 shows the performance of our approaches for discussion forum posts, compared with past work by [11]. Note that [12] do not report performance

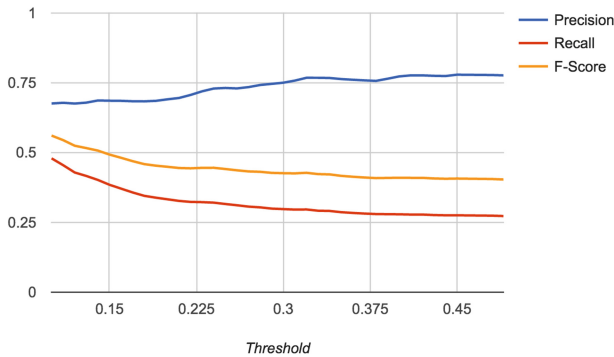


Fig. 1. Determining optimal value of threshold; Tweets, word2vec, All-words approach

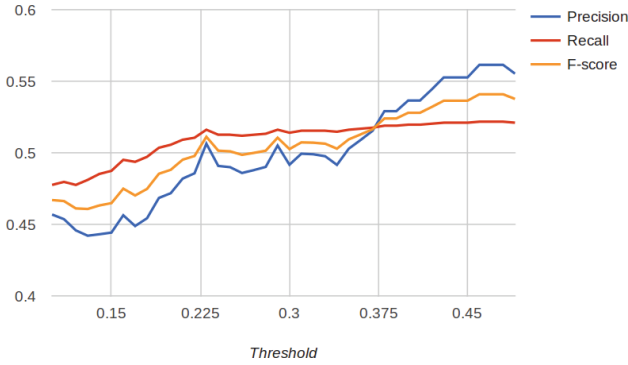


Fig. 2. Determining optimal value of threshold; Discussion Forum posts, word2vec, All-words approach

Table 1. Results of our approach for dataset of tweets by Riloff et al. (2013), compared with best reported values (Joshi et al. (2015) and Riloff et al. (2013)) on the same dataset

		P	R	F
Riloff et al. (2013)		62	44	51
Joshi et al. (2015)		77	51	61
Similarity	T	P	R	F
All-Words Approach				
Word2Vec	0.11	67.85	45.51	54.48
WordNet	0.11	67.68	74.84	71.08
Incongruous words-only Approach				
Word2Vec	0.42	68.00	77.64	72.50
WordNet	0.12	82.77	77.87	80.24

on this dataset and are hence, not included in this table. Figure 2 shows how optimal threshold is determined. Note that similar trends are observed for other cases as well. In this case, our approaches do not perform as well as the past reported value in [11]. Also, unlike the tweets, the Incongruous words-only approach results in a degradation as compared to all-words approach, for discussion forum posts. This shows that while our approach works for short text (tweets), it does not work for long text (discussion forum posts). This is because the average length of discussion forum posts is higher than that of tweets. As a result, the all-words approach or even Incongruous words-only approach may introduce similarity comparison with irrelevant words (*‘man’* and *‘woman’* in the example in Sect. 3).

Table 2. Results of our approach for dataset of discussion forum posts by Walker et al. (2012), compared with best reported value on the same dataset

		P	R	F
Joshi et al. (2015)		48.9	92.4	64
Similarity	T	P	R	F
All-Words Approach				
Word2Vec	0.48	56.14	52.17	54.08
WordNet	0.27	45.12	47.68	46.37
Incongruous words-only Approach				
Word2Vec	0.36	37.04	47.48	41.61
WordNet	0.15	42.69	48.18	45.27

Table 3. Two-fold cross-validation performance of our approaches for the tweets dataset; Best-T values in parentheses are optimal thresholds as obtained for the two folds

		P	R	F
Riloff et al. (2013)		62	44	51
Joshi et al. (2015)		77	51	61
Similarity Metric	Best-T	P	R	F
All words				
Word2Vec	(0.1, 0.1)	67.68	47.96	56.12
WordNet	(0.1, 0.1)	68.83	76.93	72.66
Incongruous words-only				
Word2Vec	(0.42, 0.1)	63.92	77.64	70.09
WordNet	(0.14, 0.12)	82.81	77.91	80.28

Table 4. Two-fold cross-validation performance of our approaches for the discussion forum posts dataset; Best-T values in parentheses are optimal thresholds as obtained for the two folds

		P	R	F
Joshi et al. (2015)		48.9	92.4	64
Similarity Metric	Best-T	P	R	F
All words				
Word2Vec	(0.48, 0.48)	56.20	52.17	54.10
WordNet	(0.37, 0.46)	43.13	48.04	45.45
Incongruous words-only				
Word2Vec	(0.19,0.25)	36.48	47.41	41.23
WordNet	(0.15,0.12)	28.34	48.04	35.33

6.2 Two-Fold Cross-Validation

Tables 3 and 4 show the two-fold cross-validation performance in case of tweets and discussion forum posts respectively. In each of the cases, past work that reports results on the same dataset is also mentioned: [12] and [11] report performance on the tweets dataset while [11] do so on the discussion forums dataset. The optimal values of threshold for the two folds are also reported since they cannot be averaged. Table 3 shows that the incongruous words-only approach outperforms past work and the all words approach. The best performance is 80.28% when incongruous words-only approach and WordNet similarity are used. Thus, in the case of tweets, our approaches perform better than past reported values.

Table 4 shows the corresponding values for the discussion forum posts. Unlike tweets, both our approaches do not perform as well as past reported values. The reported value of F-score is 64% while our approaches achieve a best F-score of 54.10%. This is likely because discussion forum posts are longer than tweets and hence, the set of candidate incongruous words is larger. This negative observation, in combination with the observation in case of tweets above, is an indicator of how the set of candidate incongruous words is a crucial parameter of the success of our approaches.

7 Discussion

Since our approaches perform well for short text like tweets but not for long text such as discussion forum posts, choosing the right set of candidate positions appears to be crucial for the success of the proposed technique. The Incongruous words-only is a step in that direction, but we observe that it is not sufficient in case of discussion forum posts. Hence, in this section, we consider an oracle case: the exact incongruous word case. This is the case where the exact incongruous word is known. Hence, we now compare our all-words approach with an ‘*exact incongruous word*’ approach, when the exact incongruous word is known. In this case, we do not iterate over all word positions but only the position of the incongruous word. For the purpose of these experiments, we use the dataset by [22]. Their dataset consists of a word, a tweet containing the word and the sarcastic/non-sarcastic label. In case of sarcastic tweets, the word indicates the specific incongruous word. Table 5 compares the all-words approach with the only incongruous word approach. We observe that the F-score increases from 55.43% to 63.42% when the exact incongruous word is known. This shows that

Table 5. Performance of the all-words approach versus the situation when the exact incongruous word is known

Approach	T	P	R	F
All-words	0.29	55.07	55.78	55.43
Oracle	0.014	59.13	68.37	63.42

our approaches can be refined further to be able to zone in on a smaller set of candidate incongruous words.

It is never possible to know the exact incongruous word in a sentence. Therefore, future approaches that follow this line of work would need to work towards reducing the set of candidate incongruous words.

8 Error Analysis

Some errors made by our approaches are due to the following reasons:

1. **Absence of WordNet senses:** For a certain input sentence, the word ‘*cottoned*’ is returned as the most likely word for a position. However, no sense corresponding to the word exists in WordNet, and so the word is ignored.
2. **Errors in sentence completion:** The sarcastic sentence ‘*Thank you for the input, I’ll take it to heart*⁷’ is incorrectly predicted as non-sarcastic. For the position where the word ‘*input*’ is present, the expected word as returned by context2vec is ‘*message*’.

9 Conclusion and Future Work

This paper describes how sentence completion can be used for sarcasm detection. Using context2vec, a sentence completion toolkit, we obtain the expected word at a given position in a sentence, and compute the similarity between the observed word at that position and the expected word. Since the position of the incongruous (observed) word may not be known, we consider two approaches: (a) All-words approach in which context2vec is invoked for all content words, (b) Incongruous words-only approach where context2vec is invoked only for 50% most incongruous words. We present our experiments on two datasets: tweets and book snippets, and for two similarity measures: word2vec similarity, and WordNet similarity. Our approach outperforms past reported work for tweets but not for discussion forum posts, demonstrating that sentence completion can be used for sarcasm detection of short text. Finally, we validate the benefit of our approach for an oracle case where the exact incongruous word is known. Our approach results in a 8% higher F-score as compared to the all-words approach. Our error analysis shows that absent WordNet senses and errors in sentence completion results in errors by our approach.

Our findings set up the promise of sentence completion for sarcasm detection. This work can be extended by incorporating the current technique as a set of features for a statistical classifier. Since our approaches do not perform well for discussion forum posts, our approach must be refined to arrive at a good subset of candidate incongruous words.

⁷ This tweet is labeled as sarcastic in the dataset by [12].


References

1. Tsur, O., Davidov, D., Rappoport, A.: ICWSM-a great catchy name: semi-supervised recognition of sarcastic sentences in online product reviews. In: ICWSM (2010)
2. Reyes, A., Rosso, P., Veale, T.: A multidimensional approach for detecting irony in twitter. *Lang. Res. Eval.* **47**(1), 239–268 (2013)
3. Joshi, A., Tripathi, V., Patel, K., Bhattacharyya, P., Carman, M.: Are word embedding-based features for sarcasm detection? In: EMNLP (2016)
4. Khattri, A., Joshi, A., Bhattacharyya, P., Carman, M.J.: Your sentiment precedes you: using an author’s historical tweets to predict sarcasm. In: WASSA, p. 25 (2015)
5. Veale, T., Hao, Y.: Detecting ironic intent in creative comparisons. In: ECAI, vol. 215, pp. 765–770 (2010)
6. Maynard, D., Greenwood, M.A.: Who cares about sarcastic tweets? investigating the impact of sarcasm on sentiment analysis. In: LREC (2014)
7. Gibbs, R.W.: *The Poetics of Mind: Figurative Thought, Language, and Understanding*. Cambridge University Press, New York (1994)
8. Ivanko, S.L., Pexman, P.M.: Context incongruity and irony processing. *Discourse Process.* **35**(3), 241–279 (2003)
9. Zweig, G., Burges, C.J.: The microsoft research sentence completion challenge. Technical Report MSR-TR-2011-129, Microsoft. Technical report (2011)
10. Melamud, O., Goldberger, J., Dagan, I.: context2vec: learning generic context embedding with bidirectional LSTM. In: CONLL, pp. 51–61 (2016)
11. Joshi, A., Sharma, V., Bhattacharyya, P.: Harnessing context incongruity for sarcasm detection. In: ACL-IJCNLP, vol. 2, pp. 757–762 (2015)
12. Riloff, E., Qadir, A., Surve, P., De Silva, L., Gilbert, N., Huang, R.: Sarcasm as contrast between a positive sentiment and negative situation. In: EMNLP, pp. 704–714 (2013)
13. Rajadesingan, A., Zafarani, R., Liu, H.: Sarcasm detection on twitter: a behavioral modeling approach. In: ICWSM. ACM, pp. 97–106 (2015)
14. Wallace, B.C., Choe, D.K., Charniak, E.: Sparse, contextually informed models for irony detection: exploiting user communities, entities and sentiment. *ACL* **1**, 1035–1044 (2015)
15. Wang, Z., Wu, Z., Wang, R., Ren, Y.: Twitter sarcasm detection exploiting a context-based model. In: Wang, J., Cellary, W., Wang, D., Wang, H., Chen, S.-C., Li, T., Zhang, Y. (eds.) WISE 2015. LNCS, vol. 9418, pp. 77–91. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-26190-4_6
16. Joshi, A., Tripathi, V., Bhattacharyya, P., Carman, M.: Harnessing sequence labeling for sarcasm detection in dialogue from TV series ‘friends’. In: CoNLL, p. 146 (2016)
17. Silvio, A., Wallace, B.C., Lyu, H., Silva, P.C.M.J.: Modelling context with user embeddings for sarcasm detection in social media. In: CoNLL 2016, p. 167 (2016)
18. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space, arXiv preprint [arXiv:1301.3781](https://arxiv.org/abs/1301.3781) (2013)
19. Liu, Q., Jiang, H., Wei, S., Ling, Z.-H., Hu, Y.: Learning semantic word embeddings based on ordinal knowledge constraints. In: ACL-IJCNLP (2015)

20. Walker, M.A., Tree, J.E.F., Anand, P., Abbott, R., King, J.: A corpus for research on deliberation and debate. In: LREC, pp. 812–817 (2012)
21. Pedersen, T., Patwardhan, S., Michelizzi, J.: Wordnet:: similarity: measuring the relatedness of concepts. In: Demonstration Papers at HLT-NAACL. Association for Computational Linguistics 2004, pp. 38–41 (2004)
22. Ghosh, D., Guo, W., Muresan, S.: Sarcastic or not: word embeddings to predict the literal or sarcastic meaning of words. In: EMNLP (2015)



Detecting Computer-Generated Text Using Fluency and Noise Features

Hoang-Quoc Nguyen-Son¹(✉)  and Isao Echizen^{1,2}

¹ National Institute of Informatics, Tokyo, Japan
{nshquoc, iechizen}@nii.ac.jp

² Graduate University for Advanced Studies, Hayama, Kanagawa, Japan

Abstract. Computer-generated text plays a pivotal role in various applications, but the quality of the generated text is much lower than that of human-generated text. The use of artificially generated “machine text” can thus negatively affect such practical applications as website generation and text corpora collection. A method for distinguishing computer- and human-generated text is thus needed. Previous methods extract fluency features from a limited internal corpus and use them to identify the generated text. We have extended this approach to also estimate fluency using an enormous external corpus. We have also developed a method for extracting and distinguishing the noises characteristically created by a person or a machine. For example, people frequently use spoken noise words (*2morrow, wanna*, etc.) and misspelled ones (*comin, hapy*, etc.) while machines frequently generate incorrect expressions (such as untranslated phrases). A method combining these fluency and noise features was evaluated using 1000 original English messages and 1000 artificial English ones translated from Spanish. The results show that this combined method had the highest accuracy (80.35%) and the lowest equal error rate (19.44%) compared with one of state-of-the-art methods, which uses syntactic parser. Moreover, experiments using texts in other languages produced similar results, demonstrated that our proposed method works consistently across various languages.

Keywords: Computer-generated text · Fluency feature
Noise features · Spoken word · Misspelled word · Untranslated word

1 Introduction

Computer-generated text plays a major role in modern life. Various applications generate such text (e.g., sentence simplification [13], headline generation [11], and conversation creation [6]). Other automatic services use such text as key components in their systems such as question answering systems [12] and chatbot systems [4]. Moreover, such text is also key component in voice response systems (Apple Siri, Microsoft Cortana, Google Assistant, etc.).

Although various methods are available for generating text, the quality of the generated text is still much lower than that of human-generated text. Therefore,

the use of computer-generated text can negatively affect practical applications. Its use can particularly affect the readability of web pages, and the quality of a crawled text corpus is generally degraded if it contains computer-generated text. A method for determining whether a person or a computer created the text in question is thus needed.

Numerous researchers have addressed the problem of detecting computer-generated text. Since such text is generally less natural than human-generated text, previous methods tend to focus on measuring the fluency of the text. There are two main approaches to this measurement. In one approach, the *salad phenomenon*, which makes text unnatural, is used. For example, the method proposed by Arase and Zhou measures the phenomenon by quantifying the fluency of continuous words by using an N -gram model and discontinuous words by using sequence pattern mining [1]. In the other approach, the fluency features are extracted using the properties of *parsing trees* [3, 7]. However, the properties of complex parsing structures cannot be exploited. The performance of these previous methods is limited due to the use of a restricted internal corpus. This reduces the effectiveness of these methods when they are applied to other corpora.

Human- and computer-generated texts can be distinguished by using special words that we call *noise*. Human noise includes words more probably written by a person than by a computer due to use of spoken language spellings (e.g., *tmr*, *2 day*) and misspellings (*hapy*, *withh*, etc.). Computer noise includes unexpectedly occurring text such as untranslated phrases.

We have developed a method that identifies computer-generated text on the basis of fluency and noise features. Our main contributions are listed below:

- We have developed a method for quantifying the fluency features of text. It measures the frequency of phrases not only in a limited internal corpus but also in a large external one.
- We propose identifying and classifying two kinds of noise: human noise and computer noise. Human noise includes written forms of spoken words (e.g., *isn't*, *thanksto*) and misspelled words (*moiton*, *comming*, etc.). The other noise is generated by computer errors such as untranslated words.
- We have developed a method for detecting spoken words by using an extended lexical spoken dictionary with slang words (e.g., *coulda*, *kinda*) and a short form list (*don't*, *he's*, etc.).
- We propose using the minimum edit distance to extract a feature for classifying misspellings and untranslated text.
- We have developed a method that uses both fluency and noise features to distinguish computer-generated text from human-generated text.

We evaluated our method for distinguishing text by using 1000 human-generated English messages and 1000 computer-generated English messages translated from Spanish by Google¹. Our method had higher accuracy (80.35%) and a lower equal error rate (19.44%) than one state-of-the-art method using

¹ <https://translate.google.com/>.

features extracted from a parsing tree [7]. Moreover, its performance was similar in evaluations using texts in other languages (Dutch, French, and Italian). This demonstrates that the proposed method performs consistently for various languages.

In Sect. 2 we review related work. Fluency and noise features are presented in Sects. 3 and 4, respectively. Their combined use is described in Sect. 5. The evaluation is described and discussed in Sect. 6. Section 7 summarizes the key points and mentions future work.

2 Related Work

Computer-generated text detection has been addressed by numerous researchers. Since human-generated text is usually more natural than computer-generated text, most previous methods quantify the fluency of the target text. There are two main approaches to evaluating text fluency. One is based on the *salad phenomenon* [1], and the other is based on the properties of a *parsing tree* [3, 7]. Several of the main methods of each approach are summarized below.

2.1 Salad Phenomenon

The salad phenomenon [1] describes the situation in which “each phrase in a sentence is semantically and syntactically correct but becomes incorrect when combined with other phrases in the sentence.” Arase and Zhou [1] proposed a method for identifying the phenomena of continuous words and discontinuous words. Language models with N -gram features are used to quantify the continuous words phenomenon while sequential pattern mining is used to quantify the discontinuous words phenomenon.

2.2 Parsing Tree

In the method proposed by Chae and Nenkova [3], features are extracted from a parsing tree and used to identify computer-generated messages. It is based on the assumption that computer-generated messages generally have a different parse tree depth than human-generated ones. Since the computer usually cannot understand the whole meaning and structure of the original message, the translated message may contain several fragment tags. Moreover, the density of key elements also differs between the two types of messages. Phrase type density, average phrase length, and phrase type rate are used to measure text density.

The method proposed by Li et al. [7] also extracts information from a parsing tree, and the density of the main components (determiners, quantifiers, pronouns, prepositions, functional marks, auxiliary verbs, conjunctions, and pronouns) in the text is quantified. They assume that a parsing tree of human translation is more balanced than a human one and thus proposed using several features (including the parsing tree depth and maximal noun phrase length in branches on the tree) to estimate the balance.

While previous methods extract fluency features from a limited internal corpus, our method extracts them from an external corpus (Web 1T 5-gram [2]). The extracted features are presented in Sect. 3. Our method also extracts noise features from spoken, misspelled, and untranslated words. These words are normally generated by either a person or a computer. These features are described in Sect. 4.

3 Fluency Features

Our proposed scheme for extracting fluency features (Fig. 1) comprises two steps.

- **Step 1a (Calculate word features)**: Each phrase of input text t is extracted and used to calculate word features W using a N -gram model ($N = 1$ to 3).
- **Step 1b (Calculate frequency features)**: The frequency of each phrase in text t is quantified and used to create features F using the Web 1T 5-gram corpus [2]. This step creates five features, $F = \{f_1, f_2, f_3, f_4, f_5\}$, corresponding to the lengths of the phrases (length = total number of words in phrase).

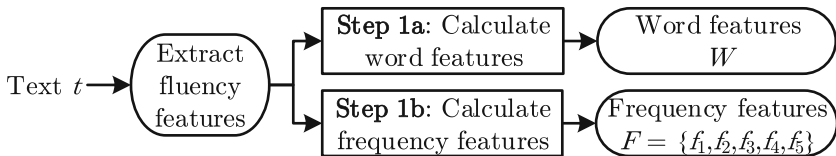


Fig. 1. Proposed scheme for extracting fluency features.

The details of each step are described below using two example messages. The computer-generated text was translated from Spanish into English using Google Translate.

t_H (human-generated text): “*Realizing all my eggs were in one basket. Taking them out one at a time. This will be fun.*”

t_M (computer-generated text): “*Euguenio mena I forget that weon we give to eat in the U xD.*”

3.1 Calculate Word Features (Step 1a)

Since human-generated messages are generally more fluent than computer-generated messages, the cohesion of successive words in text t in human-generated text is generally better than in computer-generated text. This cohesion is quantified using a commonly used N -gram model. We use a standard N -gram model with N from 1 to 3 to measure features W . This model evaluates the occurrence of continuous words in the text in an internal corpus containing those words. Several word features were extracted from messages t_H and t_M :

$W(t_H) = \{Realizing, all, \dots, Realizing\ all, all\ my, \dots, Realizing\ all\ my, all\ my\ eggs, \dots\}$

$W(t_M) = \{Euguenio, mena, \dots, Euguenio\ mena, mena\ I, \dots, Euguenio\ mena\ I, mena\ I\ forget, \dots\}$.

3.2 Calculate Frequency Features (Step 1b)

Another way to quantify the fluency of a message is to use Web 1T 5-gram [2], a huge external corpus. Sentences are separated from text t using the CoreNLP library [8]. The phrases with lengths from one to five successive words in each sentence are then calculated their frequencies in the corpus. This step creates five features, $F = \{f_1, f_2, f_3, f_4, f_5\}$, corresponding to the lengths. Each feature is an average frequency of possible phrases in text t .

For example, the five features of text t_M are calculated below. Because t_M contains an unexpected word “Euguenio,” the $Fr(Euguenio)$, $Fr(Euguenio\ mena)$, $Fr(Euguenio\ mena\ I)$, $Fr(Euguenio\ mena\ I\ forget)$, and $Fr(Euguenio\ mena\ I\ forget\ that)$ are equal 0. This affects the five features. Therefore, the features are useful for detecting computer-generated text.

$$f_1 = \frac{Fr(Euguenio) + \dots + Fr(xD)}{14} = \frac{0 + \dots + 1683551}{14} = 2.3E10$$

$$f_2 = \frac{Fr(Euguenio\ mena) + \dots + Fr(U\ xD)}{13} = \frac{0 + \dots + 0}{13} = 1.3E8$$

$$f_3 = \frac{Fr(Euguenio\ mena\ I) + \dots + Fr(the\ U\ xD)}{12} = \frac{0 + \dots + 0}{12} = 4.1E5$$

$$f_4 = \frac{Fr(Euguenio\ mena\ I\ forget) + \dots + Fr(eat\ in\ the\ U\ xD)}{11} = 4.7E3$$

$$f_5 = \frac{Fr(Euguenio\ mena\ I\ forget\ that) + \dots + Fr(eat\ in\ the\ U\ xD)}{10} = 0$$

4 Noise Features

Our proposed scheme for extracting noise features (Fig. 2) comprises four steps.

- **Step 1 (Extract set of candidate noises)**: Extract candidate noises in text t . A word is considered to be noise if it is not a name (person name, organization name, or location name). Hyperlinks, hashtags, tags, and punctuation marks are not also considered to be noise.
- **Step 2 (Extract set of spoken noise words)**: Extract candidate noise words in text t if they are written in spoken forms (e.g., *tmr*, *2 day*). These words are placed in spoken set S . The number of elements in set S is calculated and used as the spoken noise feature.
- **Step 3 (Extract set of unexpected noise words)**: Compare words remaining in text t with those in a standard dictionary to identify the set of unexpected noise words E . Set E contains noise words not in the dictionary.

- **Step 4 (Calculate minimum edit distance of unexpected noise words):** Compare expected noise word e_i with each word in the standard dictionary to calculate minimum edit distances $D_i = \{d_{i_1}, d_{i_2}, \dots, d_{i_n}\}$. The minimum number of elements u_i of D_i represents unexpected noise feature \bar{U} . This feature is used to determine whether the noise is a word misspelled by a person or a word by not translated by a computer.

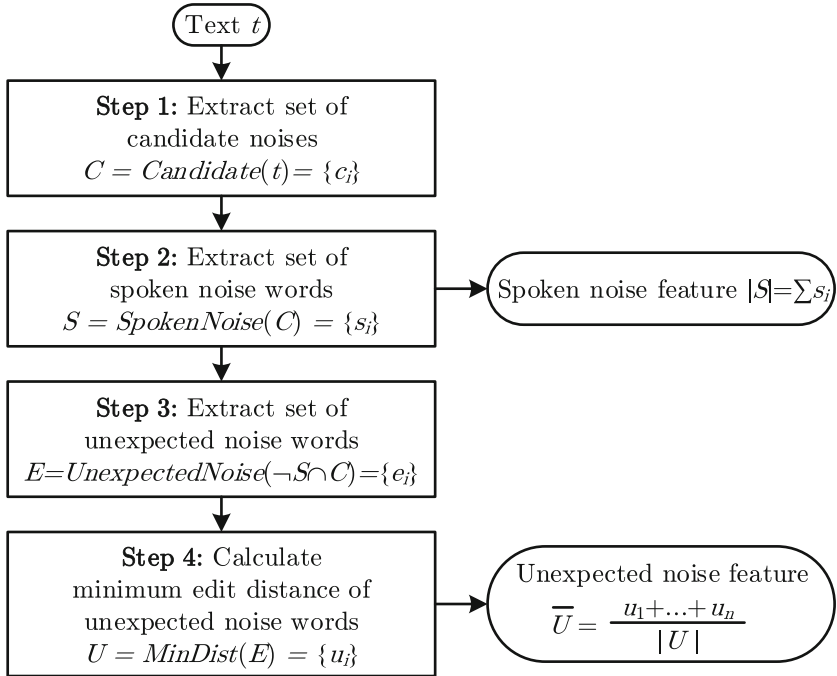


Fig. 2. Proposed scheme for extracting noise features.

Three example text messages are used in the next section. The computer-generated message was translated from Spanish into English using Google Translate.

t_{H_1} (human-generated text): “#codysimpson Pleasee do a tour in england! Shout out pleaaase!!”

t_{H_2} (human-generated text): “RT @BGR_Tycon: A #goodwoman doesnt cheat cuz her man cheated. She stands by him an inspires him to be a better man”

t_{M_1} (computer-generated text): “@Los40 geniooo pass me hey soul sister please! So we started the previous in dubai tortuguitas!”

4.1 Extract Set of Candidate Noises (Step 1)

Text t can contain proper names (e.g., persons, organizations, locations). Although they are out-of-dictionary words, they are not noise in the texts. Therefore, these words in t should be ignored before noise is detected. The Stanford Named Entity Recognizer [8] is used to detect proper names in texts.

Moreover, text t may contain certain words that have a specific meaning. For example, online social networks messages often contain hyperlinks, hashtags, and tags that often start with “*http*”, “*#*”, and “*@*”, respectively. They can be detected as regular expressions. Punctuation marks are removed to create set C of candidate noises. The sets C_{H_1} , C_{H_2} , and C_{M_1} are created from t_{H_1} , t_{H_2} , and t_{M_1} , respectively:

$$C_{H_1} = \text{Candidate}(t_{H_1}) = \{\textit>Pleasee, do, a, tour, in, Shout, out, pleaase}\}$$

$$C_{H_2} = \text{Candidate}(t_{H_2}) = \{\textit>RT, A, doesnt, cheat, cuz, her, man, cheated, She, stands, by, him, an, inspires, him, to, be, a, better, man}\}$$

$$C_{M_1} = \text{Candidate}(t_{M_1}) = \{\textit>geniooo, pass, me, hey, soul, sister, please, So, we, started, the, previous, in, dubai, tortuguitas}\}$$

4.2 Extract Set of Spoken Noise Words (Step 2)

Text t can contain spoken words, which may be written in a shortened form. These words commonly occur in online social networking messages (e.g. “*2mr*”, “*g to meet u*”) or communication passages (such as “*cuz*” and “*Ah*”). These noises are generally used only in human languages and can be identified using a spoken language dictionary. We used the one created by Bo Han et al. [5]. It contains 41,181 spoken words extracted using a statistic method. We also extend the dictionary with slang words (e.g., *gotta*, *woulda*) and short form words (*you’re*, *haven’t*, etc.). Each candidate noise word C filtered in Step 1 is matched against the extended dictionary items to determine whether it is a spoken word s_i . The total number of extracted words $|S|$ is used as the spoken noise feature.

Spoken noises S_i are extracted from the candidate noises C_i shown below. Evaluation results showed that the number of spoken words for human-generated text is frequently greater than that for machine-generated text.

$$S_{H_1} = \text{SpokenNoise}(C_{H_1}) = \{\textit>Pleasee, pleaase}\}$$

$$|S_{H_1}| = 2$$

$$S_{H_2} = \text{SpokenNoise}(C_{H_2}) = \{\textit>cuz}\}; |S_{H_2}| = 1$$

$$S_{M_1} = \text{SpokenNoise}(C_{M_1}) = \{\}; |S_{M_1}| = 0$$

4.3 Extract Set of Unexpected Noise Words (Step 3)

The words remaining ($\neg S \cap C$) after Step 2 are used to extract unexpected noise words. These words are not in a standard dictionary. We use the Oxford Advanced Learner’s Dictionary² for this. The dictionary contains 74,075 non-duplicate English items including nouns, adjectives, and adverbs.

² <http://www.oxfordlearnersdictionaries.com/>.

Since a word can have various forms (e.g., “go,” “goes,” and “went”), each word in the remaining words should be normalized. Therefore, we normalize the words by their lemmas (e.g., “zeroes” to “zero”). The Stanford CoreNLP [8] is used to determine the lemmas. Unexpected noise words E are extracted from the remaining words:

$$\begin{aligned} E_{H_1} &= UnexpectedNoise(\neg S_{H_1} \cap C_{H_1}) = \{\} \\ E_{H_2} &= UnexpectedNoise(\neg S_{H_2} \cap C_{H_2}) = \{doesnt\} \\ E_{M_1} &= UnexpectedNoise(\neg S_{M_1} \cap C_{M_1}) = \{geniooo, tortuguitas\} \end{aligned}$$

4.4 Calculate Minimum Edit Distance of Unexpected Noise Words (Step 4)

Unexpected noise words E are regularly produced by people and computers. People often create noises through typing mistakes. For example, the unexpected word “doesnt” in text E_{H_2} is the result of a missed apostrophe {’}. On the other hand, the unexpected noises produced by computer are often the result of a translation error. Words in the original text that cannot be understood by the computer become noise words in t .

Three common cases of misspellings by people include missing letters (*comin*), adding extra letters (*comming*), and switching the positions of letters (*commign*). Therefore, the minimum edit distance (MED) between a misspelled word and the corresponding original word is small. In contrast, an word not translated by the computer (such as *tortuguitas* in E_{M_1}) is generally much difference than dictionary words. The MED between an untranslated word and dictionary items is therefore usually larger. Therefore, the MED can be used to distinguished unexpected noise words as either human-generated or machine-generated.

We calculate the MED between each unexpected noise word e_i and Oxford dictionary words. The MED is measured using a dynamic programming algorithm [10]. The shortest MED u_i is used to represent an unexpected noise word. The average u_i is used as unexpected noise feature \bar{U} . The values of u_i for unexpected noise E_i and noise feature \bar{U} are shown below. The \bar{U} for computer-generated text is usually greater than that for human-generated text.

$$\begin{aligned} \bar{U}_{H_1} &= MinDis(E_{H_1}) = \{\}; \\ \bar{U}_{H_1} &= 0 \\ \bar{U}_{H_2} &= MinDis(E_{H_2}) = \{MinDis(doesnt)\} = \{1\}; \\ \bar{U}_{H_2} &= 1 \\ \bar{U}_{M_1} &= \{MinDis(geniooo), MinDis(tortuguitas)\} = \{3, 4\} \\ \bar{U}_{M_1} &= \frac{3+4}{2} = 3.5 \end{aligned}$$

5 Combination of Fluency and Noise Features

Our proposed scheme for distinguishing computer- and human-generated (Fig. 3) text comprises two steps.

- **Step 1a (Extract fluency features)**: Extract fluency features in text t including word features W and five frequency features $F = \{f_1, f_2, f_3, f_4, f_5\}$ using the steps in Sect. 3.
- **Step 1b (Extract noise features)**: Extract noise features in text t and create spoken noise feature $|S|$ and unexpected noise feature \bar{U} as described in Sect. 4.
- **Step 2 (Detect translated text)**: Use fluency and noise features to determine whether text t was written by a person or translated by a computer.

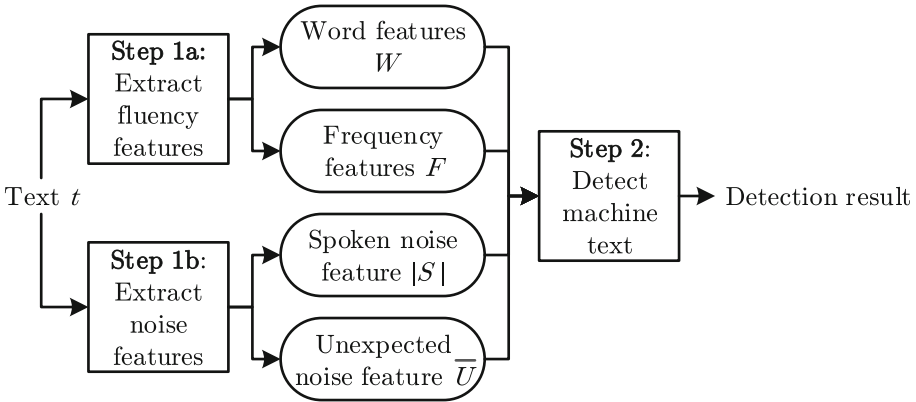


Fig. 3. Proposed scheme for distinguishing computer- and human-generated text.

Detect translated text in detail (Step 2): The features are extracted in Steps 1a and 1b. They are then combined and run on a classifier. The classifier used is the one with the best performance among commonly used machine learning algorithms (naive Bayes, logistic regression, and sequential minimal optimization (SMO)), which run on individual features. The SMO algorithm was used here.

6 Evaluation

6.1 Individual Features

The Tweet 2011 corpus [9] was used to evaluate the proposed method. It contains about 16 million tweets in various languages. On the basis of language popular, we used English and Spanish messages for our main experiments. We randomly extracted 1000 Spanish and 1000 English messages from the corpus. The English and Spanish messages were identified by using a language detection tool³. The English messages were considered to be human-generated texts. The Spanish

³ <https://github.com/shuyo/language-detection>.

messages were translated into English using Google Translate. The translated messages were considered to be computer-generated texts.

The 2000 messages were evaluated using the individual fluency and noises features. Each feature was tested using commonly used machine learning algorithms (naive Bayes, logistic regression, and SMO) to create classifiers with 10-fold validation. The classifiers were then used to distinguish the computer- and human-generated texts. Table 1 shows the accuracy (Acc) and equal error rate (EER) metrics.

Table 1. Evaluation of individual features.

Individual features		Naive bayes		Logistic regression		SMO	
		Acc	EER	Acc	EER	Acc	EER
Fluency	Word W	75.40%	24.90%	76.75%	23.20%	78.45%	21.23%
	Frequency F	56.75%	40.50%	61.35%	38.50%	59.05%	28.09%
Noise	Spoken noise $ S $	62.90%	39.15%	63.90%	40.24%	61.35%	41.19%
	Unexpected noise \bar{U}	59.70%	43.66%	59.50%	44.10%	59.75%	22.69%

SMO had the best performance for word features W —it had the highest accuracy (78.45%) and the lowest equal error rate (21.23%). Therefore, SMO was used to create the classifier was for the subsequent testing. The ongoing improvement in computer translation is reducing the noise in computer-generated texts. This has reduced the effectiveness of using only individual frequency features F or unexpected noise feature \bar{U} . Moreover, some messages may not contain spoken noises, so the spoken noise feature $|S|$ is less effective in distinguishing computer-generated text from human-generated text. However, the combination of these individual features improves the performance, as described in below.

6.2 Combination

The individual features were classified into two groups: fluency and noise. The effectiveness of these groups was then evaluated for the same 2000 messages. The features of the two groups were then merged and run through the SMO algorithm to create a final classifier. The results were compared with Li et al. method [7], which is based on syntactic parser. We choose this method to re-implement because of its independence. On the other hand, other previous methods need annotators to evaluate training data. For example, Chae and Nenkova’s method [3] need assessors to quantify the level of fluency for their gold-standard dataset.

As shown in Table 2, accuracy was 78.90% when word features W was combined with the five frequency features extracted from the huge external corpus. This shows that the external efficiently support for the internal corpus. The highest accuracy (80.35%) and lowest equal error rate (19.44%) was obtained using

Table 2. Evaluation of combination of individual features.

Method	Accuracy	Equal error rate
Li et al. [7]	63.85%	35.16%
Word features W	78.45%	21.23%
Fluency features (W and F)	78.90%	20.69%
Noise features ($ S $ and \bar{U})	64.30%	39.80%
Combination	80.35%	19.44%

a combination of fluency and noise features. This shows the importance of using noise features integrated with fluency features for detecting computer-generated texts.

6.3 Other Languages

Similar testing was done using other languages (Dutch, French, and Italian). Again, 1000 messages in each language were randomly chosen for evaluation using word features and a combination of fluency and noise features. These messages were also translated into English using Google Translate and considered to be computer-generated text. The 1000 English messages used in the previous testing were used as the human-generated text.

Table 3. Evaluation using other languages.

Language	Method	Accuracy	Equal error rate
Dutch	Li et al. [7]	57.70%	41.89%
	Combination	76.35%	24.83%
French	Li et al. [7]	62.80%	36.24%
	Combination	78.35%	23.73%
Italian	Li et al. [7]	73.35%	26.00%
	Combination	82.25%	19.02%

As shown in Table 3, performance using the combination was similar for the other languages and was better than that using parsing tree of Li et al. method [7]. This indicates that the proposed method has consistent performance across various languages.

7 Conclusion

We have presented a method for detected whether text was created by a person or generated by a computer. It is based on quantifying the fluency of the text

because human-generated messages are generally more natural than computer-generated ones. Previous methods estimate the fluency by extracting features from a limited internal corpus. Our method uses an external corpus to calculate the frequency of phrases in the text. The frequency is used to improve the quantification of the fluency features.

Since human- and computer-generated texts can be identified on the basis of human-generated spoken and misspelled words and computer-generated untranslated phrases, we developed a method to identify spoken words using a lexical spoken dictionary. We classify misspelled and untranslated words using the minimum edit distance. These words are extracted as noise features.

Finally, we presented a method that combines fluency and noise features. It had better performance (accuracy = 80.35%; equal error rate = 19.44%) than a method using parsing tree. These results are for an experiment using messages translated from Spanish into English and human-generated English messages. The performance for three other languages was similar, demonstrating that our method has consistent performance across languages.

Future work includes determining the effect of the frequency features on other corpora in areas other than online social networking, improving the noise features by weighting them using the frequency of misspelled, spoken, and untranslated words, and quantifying the noise phrases in text.

Acknowledgments. This work was supported by JSPS KAKENHI Grants (JP16H06302 and JP15H01686).

References

1. Arase, Y., Zhou, M.: Machine translation detection from monolingual web-texts. In: Proceedings of the 51st Annual Meeting on Association for Computational Linguistics, pp. 1597–1607 (2013)
2. Brants, T., Franz, A.: Web 1T 5-gram version 1. Linguistic Data Consortium (2006)
3. Chae, J., Nenkova, A.: Predicting the fluency of text with shallow structural features: case studies of machine translation and human-written text. In: Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics, pp. 139–147 (2009)
4. Freitas, C., Benevenuto, F., Ghosh, S., Veloso, A.: Reverse engineering socialbot infiltration strategies in Twitter. In: Proceedings of the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, pp. 25–32 (2015)
5. Han, B., Cook, P., Baldwin, T.: Automatically constructing a normalisation dictionary for microblogs. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp. 421–432 (2012)
6. Li, J., Galley, M., Brockett, C., Gao, J., Dolan, B.: A persona-based neural conversation model. In: Proceedings of the 54th Annual Meeting on Association for Computational Linguistics, pp. 994–1003 (2016)
7. Li, Y., Wang, R., Zhai, H.: A machine learning method to distinguish machine translation from human translation. In: Proceedings of the 29th Pacific Asia Conference on Language, Information and Computation, pp. 354–360 (2015)

8. Manning, C.D., Surdeanu, M., Bauer, J., Finkel, J.R., Bethard, S., McClosky, D.: The Stanford CoreNLP natural language processing toolkit. In: Proceedings of the 52nd Annual Meeting on Association for Computational Linguistics-System Demonstrations, pp. 55–60 (2014)
9. Ounis, I., Macdonald, C., Lin, J., Soboroff, I.: Overview of the TREC-2011 microblog track. In: Proceedings of the 21st Text REtrieval Conference. vol. 32, p. 20 (2012)
10. Strube, M., Rapp, S., Müller, C.: The influence of minimum edit distance on reference resolution. In: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, pp. 312–319 (2002)
11. Sun, R., Zhang, Y., Zhang, M., Ji, D.: Event-driven headline generation. In: Proceedings of the 53rd Annual Meeting on Association for Computational Linguistics, pp. 462–472 (2015)
12. Zhou, T.C., Lyu, M.R., King, I.: A classification-based approach to question routing in community question answering. In: Proceedings of the 21st International Conference on World Wide Web, pp. 783–790. ACM (2012)
13. Zhu, Z., Bernhard, D., Gurevych, I.: A monolingual tree-based translation model for sentence simplification. In: Proceedings of the 23rd International Conference on Computational Linguistics, pp. 1353–1361 (2010)

Automatic Speech Recognition



Speaker Adaptation on Myanmar Spontaneous Speech Recognition

Hay Mar Soe Naing^(✉) and Win Pa Pa

Natural Language Processing Laboratory, UCSY, Yangon, Myanmar
{haymarsoenaing, winpapa}@ucsy.edu.mm

Abstract. This paper introduces the work on automatic speech recognition (ASR) of Myanmar spontaneous speech. The recognizer is based on the Gaussian Mixture and Hidden Markov Model (GMM-HMM). A baseline ASR is developed with 20.5 h of spontaneous speech corpus and refine it with many speaker adaptation methods. In this paper, five kinds of adapted acoustic models were explored; Maximum A Posteriori (MAP), Maximum Mutual Information (MMI), Minimum Phone Error (MPE), Maximum Mutual Information including feature space and model space (fMMI) and Subspace GMM (SGMM). We evaluate these adapted models using spontaneous evaluation set consists of 100 utterances from 61 speakers totally 23 min and 19 s. Experiments on this speech corpus show significant improvement of speaker adaptative training models and SGMM-based acoustic model performs better than other adaptative models. It can significantly reduce 3.16% WER compared with the baseline GMM model. It is also investigated that the Deep Neural Network (DNN) training on the same corpus and evaluated with same evaluation set. With respect to the DNN training, the result reaches up to 31.5% WER.

Keywords: Spontaneous speech · ASR · Myanmar · GMM
Adaptative training · DNN

1 Introduction

Automatic speech recognition (ASR) can be defined as the independent, computer-driven transcription of spoken language into human readable text in real time. Although ASR technology is not yet at the point where machines understand all speech in any acoustic environment or by any person, it is used on a day-to-day basis in a number of applications and services.

In ASR, although speech derived from read texts, news, broadcasts and other similar prompted contexts can be recognized with high accuracy, but the recognition performance decreases drastically for spontaneous speech because of the differences between the acoustic characteristics of spontaneous and read speech. Compared with controlled read speech, spontaneous speech can be characterized by varied speaking rate, filled pauses, corrections, hesitations, repetitions, partial

words, disfluencies, sloppy pronunciation [1]. Spontaneous conversation is optimized for human to human communication. The speaking rate is highly inconsistent, both within utterances, across utterances within a session, and across sessions and speakers. The articulation is highly variable. The utterance lengths vary widely and are typically longer than read speech utterances. Many speech recognition researches for Myanmar language focused on read speech utterances and there is no recognizer for spontaneous speech.

The accuracy of speech recognition degrades rapidly when there is a mismatch between the training and test conditions and also a mismatch between speakers. One approach to solve this problem is to adapt an existing speaker independent (SI) model to a speaker dependent (SD) model with some speaker specific data [2]. The idea of adaptation methods is to modify the mean vectors and covariance matrices of Gaussian distributions of the initial model using a limited amount of adaptation data to maximize the likelihood for these adapted data. The adaptation process takes transcribed data and improves the model already have. It is more robust than training and could lead to a good results even if adaptation data is small. It just improves the fit between the adaptation data and the model.

The remainder of this paper is organized as follows: Sect. 2 explains the related research and brief explanation of Myanmar spontaneous speech is described in Sect. 3. Section 4 describes the spontaneous speech data corpus which we used for our experiments. We discuss the different speaker adaptation approaches in Sect. 5. Sections 6 and 7 present the experimental setup and results. The conclusion and future work are shown in Sect. 8.

2 Related Research

In [3], the authors presented an Indonesian speech recognizer to recognize both read and spontaneous speech. They trained their model on 73 h of read speech and 43.5 min of spontaneous speech. They employed three kinds of adaptation in acoustic models; MAP, MMI and Feature-space Maximum Likelihood Linear Regression (fMLLR) on both read and spontaneous speech. The MAP adapted model is more accurate in spontaneous speech over the unadapted model and MMI adaptation. For dictation scenario, MMI adaptation performs better than the unadapted model and MAP adaptation. They also investigated that fMLLR adaptation degrades performance significantly in both scenarios. In addition, they reported that MAP and MMI adaptation is suitable when adaptation data is scarce in under-resourced languages.

Reference [4] described the Persian speech recognition system with the use of discriminative criteria for acoustic model training. They investigated that the discriminative training such as MPE and MMI performs better than the standard maximum likelihood (ML) training. According to their experiments, they used the MPE criterion to estimate the discriminative linear transforms (DLTs) for speaker adaptation. Moreover, the MPE-based DLT can improve the accuracy of recognition compared with the Maximum Likelihood Linear Regression (MLLR) adaptation.

In [5], the recognition performance of Indonesian speech is significantly degraded in recognition of spontaneous speech data. To improve the performance, they conducted many model enhancement methods by adding spontaneous data and retraining both acoustic and language models, by adapting the acoustic model based on the maximum likelihood linear regression and maximum a posteriori approach and by adapting the language model employing with linear interpolation. Their experiments show that retraining both acoustic and language models by combining spontaneous and read data is more effective than the model adaptation.

3 Myanmar Spontaneous Speech

Spontaneous voice is unplanned or unprepared voice in which rehearsal is not done and which is naturally said by the speaker. Its transcript is not well prepared in advance such as in an interview, presentation, meeting and daily conversation. Disfluency is one of the major problems in speech processing fields, especially for automatic speech recognition. It usually occurs when the speaker needs some time to think in advance and keeps audience attention, and when the speaker is nervous or confuse [5].

There are various features of spontaneous speech including filled pauses, fillers, false starts, repetitions, contracted forms, ellipsis, non-standard grammar and hesitations or silent pauses. We commonly found five kinds of disfluency form in Myanmar language. These are:

1. Filled pauses: Sound such as ah, uh, um, etc. which gives speaker time to think.
2. Fillers: Adding some words which do not carry conventional meaning but which are inserted in speech to give speaker time to plan what to say.
3. Repetitions: Repeating a word or phrase that can be emphasis or to give the speaker more time to sort out their thoughts before continuing.
4. Ungrammatical structure: They break standard grammar rules or the omission of part of a grammatical structure.
5. Contracted forms: In addition to verbal contractions, some words are shortened.

4 Speech Corpus Description

The spontaneous speech corpus was conducted for talking about the inconvenient situation in the error of ATM machine, talking about the damage and preparation by natural disaster (storm, earthquake and so on), renting car for trip, buying ticket for trip, dialogue for buying something (books, watch, clothes, computer and so on), talking about going to clinic and take medical checkup. Utterances were recorded in 16-bit mono channel WAV at a sampling rate of 16 kHz in an open environment such as Hotel lobbies and office rooms using iPhone as recording device. 52 male and 48 female speakers attended in recording

and each speaker was asked to record 50 sentences. This corpus was constructed by National Institute of Information and Communications Technology (NICT), Kyoto, Japan. In total, the spontaneous corpus consists of about 20.5 h of speech distributed over 5 K utterances. The detail of corpus and data size of training (SPONT-TR), testing (SPONT-EV) and development (SPONT-DV) data are summarized in Table 1.

Table 1. The summary of Myanmar spontaneous speech corpus

Type	# Speaker	# Utterance	Duration
SPONT-TR	100	4.8 K	19 h 41 min 31 s
SPONT-DV	62	100	25 min 10 s
SPONT-EV	61	100	23 min 19 s
Total	100	5 K	20.5 h

5 Model Adaptation Approaches

Speaker adaptation is essential to produce a speaker specific system from a speaker independent system by giving only a small amount of adaptation data. Adaptation of acoustic models are employed to reduce the recognition error rate of a generic acoustic model. Adaptation can be applied to make the more suitable model for differences in acoustic environment or the characteristic of a group of speakers.

5.1 MAP Adaptation

The Maximum A Posteriori (MAP) adaptation is used in statistical modeling and is sometimes referred to as Bayesian adaptation. The MAP estimation framework provides a way of incorporating prior knowledge about the model parameter distribution in training process, which is useful for dealing with problems posed by sparse training data for which the Maximum Likelihood (ML) approach gives inaccurate estimates. MAP can be applied in two classes of applications: parameter smoothing and model adaptation. Its main feature is to take advantage of prior information in training process, thus reducing the data needed to obtain a good speaker dependent acoustic model [6].

Let $f(X | \theta)$ be the probability density function of variable x . We estimate its parameter θ by using n samples of x , $X = \{x_1, x_2, \dots, x_n\}$. Let $g(\theta)$ be the prior distribution for θ . The probability density function of the parameter after observing X , $g(\theta | X)$ is called a posterior distribution. By Bayes Theorem,

$$g(\theta | X) = \frac{f(X | \theta) g(\theta)}{\int f(X | \theta) g(\theta) d\theta} \quad (1)$$

MAP estimation obtains the value of θ_{MAP} which gives the maximum of the posterior distribution. The θ_{MAP} is:

$$\theta_{MAP} = \operatorname{argmax}_{\theta} g(\theta | X) \quad (2)$$

$$\theta_{MAP} = \operatorname{argmax}_{\theta} f(X | \theta) g(\theta) \quad (3)$$

5.2 MMI Adaptation

Maximum Mutual Information (MMI) criterion simultaneously consider the HMMs of all classes in training phrase. The parameters of correct model are updated to enhance the contribution to the observations, while the parameters of alternative models are updated to reduce their contributions. It can provide a high discriminative ability to the HMM system. MMI adaptation [7] is a discriminative training method to estimate the new parameter of an HMM, one must maximize the numerator term and minimize the denominator term in the objective function. The goal of MMI is to maximize the mutual information between data and their corresponding labels or symbol using the expressions for entropy, conditional entropy and mutual information. This estimation requires choosing the parameter set λ to maximize the following function:

$$F_{MMIE} = \sum_{r=1}^R \log \frac{P_{\lambda}(O_r | M_{wr})P(W_r)}{\sum_{\hat{w}} P_{\lambda}(O_r | M_{\hat{w}})P(\hat{W})} \quad (4)$$

Where M_w is the HMM corresponding to the transcription W, $P(W)$ is the probability of the word sequence W as determined by the language model, and the denominator sums over each possible word sequence \hat{W} .

5.3 MPE Adaptation

Minimum Phone Error (MPE) criterion [2,4] is commonly used in discriminative criteria for acoustic model training and used for discriminative speaker adaptation. The discriminative training takes account of possible competing word hypotheses and tries to reduce the recognition errors. The optimization of discriminative MPE criterion is more complex than the standard ML criterion. The extended Baum-Welch algorithm can be used to overcome the problem of optimization. Using lattice for discriminative training can solve the computation problem and I-smoothing technique can improve the model generalization. The aim of MPE is to maximize the phone accuracy on output of recognition from given training set. The MPE objective function was defined as:

$$F_{MPE} = \sum_{r=1}^R \frac{\sum_{\hat{w}} P_{\lambda}(O_r | M^{\hat{w}})^{\kappa} P(\hat{W}) \operatorname{Rawaccuracy}(\hat{W})}{\sum_w P_{\lambda}(O_r | M^w)^{\kappa} P(W)} \quad (5)$$

Where F_{MPE} is weighted average of $\operatorname{Rawaccuracy}(\hat{W})$ over all \hat{W} . $\operatorname{Rawaccuracy}(\hat{W})$ measures the number of phones correctly transcribed in sentence w. M^w is the composite model corresponding to the word sequence w, $P(W)$ is the probability of the word sequence w and κ is the acoustic scale.

5.4 fMMI Adaptation

Discriminative training on feature transformation can be effective in recognition performance. Feature space discriminative training fMMI involves optimization of the feature transform using some discriminative training MMI which performs some form of linear transformation on the feature vectors and the transformation is optimized [8]. The transformation can be formulated as:

$$z_t = x_t + Mh_t \quad (6)$$

where x_t is the original feature, h_t is the Gaussian posterior vector computed by a GMM. M is the linear transform which is optimized for MMI objective using gradient ascent and z_t is the final feature vector.

5.5 SGMM Adaptation

Subspace Gaussian Mixture Model (SGMM) [9, 10] is an acoustic modeling approach in which all phonetic states share a common Gaussian Mixture Model structure, and the means and mixture weights vary in a subspace of the total parameter space. The model parameters are derived from a set of state dependent parameters and from a set of globally shared parameters which capture the phonetic and speaker variation. The parameterization of SGMM facilitates the acoustic model using relatively small amount of speech data as compared to the GMM. As each state in SGMM is parameterized by a small number of low dimensional state vectors v_j , it provides the modeling of a subword unit using only a relatively small number of occurrences of that unit in the training dataset. This characteristic makes the SGMM a better option when there is a paucity of data. The basic form of SGMM model can be expressed as follows:

$$p(X | j) = \sum_{i=1}^I w_{ji} \mathfrak{N}(X; \mu_{ji}, \Sigma_i) \quad (7)$$

$$\mu_{ji} = M_i v_j \quad (8)$$

$$\omega_{ji} = \frac{\exp w_i^T v_j}{\sum_{i'=1}^I \exp w_{i'}^T v_j} \quad (9)$$

where $X \in \mathfrak{R}^D$ is the feature, j is the speech state (phone in context), $v_j \in \mathfrak{R}^S$ is the state vector, I is the number of Gaussians in each state, mixture weights ω_{ji} , means μ_{ji} and full covariances Σ_i which are shared between states. The means and mixture weights are not parameters of the model. They are derived from a state specific vector v_j with the subspace dimension S typically being around the same as the feature dimension D .

6 Experimental Setup

6.1 Language Model

The MIT language modeling (MITLM) toolkit [11] is used to build 3-gram language model (LM) on the training transcription data which is segmented as word units and modified with Kneser-Ney smoothing. We trained a LM on 4.8 K sentences of transcribed data to produce the spontaneous speech language model. Language model interpolation is commonly used in combining several LMs specific to a particular style of language model are generated. Therefore, this LM is interpolated with 1.6 K sentences of daily conversation from WEB to cover a wide variation of tasks and to get the more spontaneous speech traits. Finally, we used 6.4 K sentences of textual data to build a spontaneous speech LM.

To evaluate the language models, we computed perplexity and Out of Vocabulary (OOV) rate. For the experiments, we selected 100 sentences which contain 4,346 words for evaluation and another 100 sentences for development data. It is observed that interpolating with daily conversation from WEB data can give the perplexity of 133.05 and can reduce the OOV rate from 1.93 to 1.5. In Table 2 shows the training corpus, computed perplexity and OOV rate over the same evaluation text set.

Table 2. Component LMs and interpolated LM

Corpora	# Sentences	# Words	# Perplexity	OOV rate
Transcript	4.8 K	180 K	130.5	1.93
WEB	1.6 K	58 K	401.6	12.26
Interpolate	6.4 K	238 K	133.0	1.5

6.2 Pronunciation Lexicon

For pronunciation lexicon, we used the existing work in [12] which is obtained by combining the pronunciations of all words in the Phonetically Balanced Corpus (PBC) text and standard Myanmar Language Commission (MLC) dictionary. Myanmar language is a tonal language and basically there are four tones such as low, high, creaky and checked. Moreover, the tones are indicated with diacritics or special letters in writing Myanmar scripts. This pronunciation lexicon is built with the optimal phoneme set which contains 108 phoneme units [12] that include the tone information. The vocabulary of this lexicon contains 33,576 unique words and implemented with 108 unique phonemes set for correspondent pronunciation.

6.3 Acoustic Model

As Myanmar is a tonal language, the utilization of tone information can help to improve the ASR system [12]. In this experiment, Mel Frequency Cepstral

Coefficient (MFCC) with pitch feature (MFCC (dimension = 13) + Pitch (dimension = 3)) are used to extract features from speech because the pitch is regarded as being closely related to tone patterns. This feature contains not only the fundamental frequency but also the voicing probability of current frame and F0 delta of neighboring frames. All models are trained with standard cepstral mean-variance normalized (CMVN) acoustic feature without energy and its first and second derivatives.

For our spontaneous ASR, firstly we trained a GMM of speaker independent GMM(SI) triphone by using the Linear Discriminant Analysis (LDA) and we also applied the Maximum Likelihood Linear Transform (MLLT). Secondly, we built the GMM speaker adaptive training GMM(SAT) with 3,000 leaves and 45,000 gaussians by using feature-space Maximum Likelihood Linear Regression (fMLLR) transformation on feature estimation of GMM(SI). We also implemented five kinds of speaker adaptation methods: MAP, MMI, MPE, fMMI and SGMM to compare their effectiveness in improvement of recognition accuracy on Myanmar spontaneous speech corpus. All these adaptations were directly on top of the GMM(SAT) model.

In MAP adaptation, it just did one iteration to the baseline model in the input alignment directory. It is useful for adapting a system to a specific gender, or new acoustic conditions. We used 20 tau smoothing constant in this MAP estimation. For the discriminative MMI and MPE training, we used 4 iterations of Extended Baum-Welch update. Furthermore, we also did MMI discriminative training including feature-space and model-space components (fMMI). On the iterations of training, it alternates feature-space and model-space training. We used 8 iterations in total and 4 of each type MMI and fMMI. Moreover, we trained a Universal Background Model (UBM) with 600 gaussians and applied to SGMM training with speaker vectors. We used 3000 leaves and 45000 gaussians for SGMM adaptive training.

Kaldi speech recognition toolkit is used for trainings of all acoustic models and decoding process of speech [13].

7 Experimental Result

The training set consists of 100 speakers about 20 h and development set consists of 62 speakers about 25 min of data. To evaluate the model, we computed the Word Error Rate (WER) on 100 utterances. It contained 61 speakers (28 male and 33 female) about 23 min. The whole data sets used for the experiments are shown in Table 1. The interpolated 3-gram language model is also used with a test perplexity of 133.05 and an OOV rate of 1.54 over total 4,346 words.

Speaker Independent (SI) model is obtained using the Linear Discriminant Analysis (LDA) and Maximum Likelihood Linear Transform (MLLT) criterion. The recognition accuracy of SI model can vary from speaker to speaker because of the effect of individual voice characteristics, speaking manners, styles, topics and noise. The speaker adaptive training (SAT) using feature-space Maximum Likelihood Linear Regression (fMLLR) are applied to overcome the speech variation in the utterances. From the experimental results, the speaker adaptive

training can reduce the WER in comparison with speaker independent triphone model. The speaker adaptative GMM(SAT) model gave a 3.7% reduction in WER over speaker independent GMM(SI) model as shown in Table 3.

Table 3. WER(%) of speaker independent model and speaker adaptative model

Model Type	WER (%)
GMM (SI) + LDA + MLLT	39.88
GMM (SAT) + fMLLR	36.17

We used the GMM (SAT) model as the baseline for further experiments. The main purpose of these experiments are to investigate the effectiveness of speaker adaptation on spontaneous speech recognition. We performed some experiments using the data and model adaptation approaches described in Sect. 5.

Table 4 shows the WER(%) of different speaker adaptation on same evaluation set. From these results, we can see that SGMM based acoustic model performs better than other adaptation methods and can significantly reduce 3.16% WER compared with baseline GMM(SAT) model. We also investigated that SGMM adaptation gives a 1.03% reduction in WER over MMI adaptation, 1.43% reduction in WER over MPE adaptation, 1.52% reduction in WER over fMMI and 2.79% reduction in WER over MAP adaptation.

Table 4. WER(%) of different adaptative trainings

Adaptation type	WER(%)
Baseline	36.17
GMM(SAT) + MAP	35.83
GMM(SAT) + MPE	34.47
GMM(SAT) + MMI	34.07
GMM(SAT) + fMMI	34.56
GMM(SAT) + SGMM	33.04

Since Deep Neural Network (DNN) is a current state-of-the-art acoustic modeling technique for ASR, we also explored the cross entropy criterion based DNN training on top of the baseline GMM(SAT) model. We used the same features as the GMM(SAT) and 11 frames of context windows of the fMLLR feature. Moreover, we used 5 hidden layers, each layer has 378 input units and 2,556 output units in DNN training. The evaluation set that used for GMM is conducted in DNN evaluation. In Table 5, It can be seen that the recognition performance of DNN in WER(%). According to these result, we can significantly reduce the error rate of our spontaneous speech recognition. The WER reduction can reach up to relative 12.7% with the use of DNN training.

Table 5. WER(%) of speaker adaptive GMM model and cross entropy based DNN model

Model type	WER(%)
GMM(SAT)	36.17
DNN(Cross entropy)	31.59

8 Conclusion and Future Work

In this study, we introduced a Myanmar spontaneous ASR system. We built a baseline system on a recorded 5K spontaneous speech corpus. We also investigated the different types of adaptation: MAP, MMI, MPE, fMMI, SGMM models. The SGMM improved recognition accuracy in this corpus compared with other adaptation methods. Therefore, it is concluded that SGMM adaptation is a better option when the data is paucity in case of under-resourced languages. Furthermore, We also explored the training with cross entropy criterion based DNN. The best WER of 31.5% was obtained with the DNN approach by evaluating with 100 utterances.

Future works include collecting more speech and textual data and conducting more effective methods for language model cleaning and interpolation to reduce the perplexity of the corpus. This study only focuses on the recorded spontaneous speech. There are still many interesting avenues of research to focus on more spontaneous speech such as live interview, meeting, speech and talk and so on.

Acknowledgements. We would like to thank all members of Spoken Language Communication Lab., from National Institute of Information and Communications Technology (NICT), Kyoto, Japan, for the utilization of Myanmar spontaneous speech corpus presented in this paper.

References

1. Liu, G., Lei, Y., Hansen, J.H.: Dialect identification: impact of differences between read versus spontaneous speech. In: 2010 18th European Signal Processing Conference, pp. 2003–2006. IEEE (2010)
2. Chen, L.Y., Lee, C.J., Jang, J.S.R.: Minimum phone error discriminative training for Mandarin Chinese speaker adaptation. In: INTERSPEECH, pp. 1241–1244 (2008)
3. Hoesen, D., Satriawan, C.H., Lestari, D.P., Khodra, M.L.: Towards robust Indonesian speech recognition with spontaneous-speech adapted acoustic models. *Procedia Comput. Sci.* **81**, 167–173 (2016)
4. Pirhosseinloo, S., Ganj, F.A.: Discriminative speaker adaptation in Persian continuous speech recognition systems. *Procedia Soc. Behav. Sci.* **32**, 296–301 (2012)
5. Lestari, D.P., Irfani, A.: Acoustic and language models adaptation for Indonesian spontaneous speech recognition. In: 2015 2nd International Conference on Advanced Informatics: Concepts, Theory and Applications (ICAICTA), pp. 1–5. IEEE (2015)

6. Saz, O., Vaquero, C., Lleida, E., Marcos, J.M., Canalís, C.: Study of maximum a posteriori speaker adaptation for automatic speech recognition of pathological speech. In: Proceedings of Jornadas en Tecnología del Habla (2006)
7. Vertanen, K.: An overview of discriminative training for speech recognition, pp. 1–14. University of Cambridge (2004)
8. Hsiao, R., Schultz, T.: Generalized discriminative feature transformation for speech recognition. In: INTERSPEECH, pp. 664–667 (2009)
9. Povey, D., Burget, L., Agarwal, M., Akyazi, P., Feng, K., Ghoshal, A., Glembek, O., Goel, N.K., Karafiát, M., Rastrow, A., et al.: Subspace Gaussian mixture models for speech recognition. In: 2010 IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP), pp. 4330–4333. IEEE (2010)
10. Ghalehjegh, S.H.: New Paradigms for Modeling Acoustic Variation in Speech Processing. Ph.D. thesis, McGill University (2016)
11. Hsu, B.J.P., Glass, J.R.: Iterative language model estimation: efficient data structure & algorithms. In: INTERSPEECH, pp. 841–844 (2008)
12. Naing, H.M.S., Hlaing, A.M., Pa, W.P., Hu, X., Thu, Y.K., Hori, C., Kawai, H.: A Myanmar large vocabulary continuous speech recognition system. In: 2015 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA), pp. 320–327. IEEE (2015)
13. Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., Hannemann, M., Motlicek, P., Qian, Y., Schwarz, P., et al.: The kaldi speech recognition toolkit. In: IEEE 2011 Workshop on Automatic Speech Recognition and Understanding, No. EPFL-CONF-192584. IEEE Signal Processing Society (2011)



Exploring the Effect of Tones for Myanmar Language Speech Recognition Using Convolutional Neural Network (CNN)

Aye Nyein Mon^{1(✉)}, Win Pa Pa¹, and Ye Kyaw Thu²

¹ Natural Language Processing Laboratory, University of Computer Studies,
Yangon, Myanmar

{ayenyeinmon, winpapa}@ucsu.edu.mm

² Artificial Intelligence Laboratory, Okayama Prefectural University,
Okayama, Japan
ye@c.oka-pu.ac.jp

Abstract. Tone information is very helpful to improve automatic speech recognition (ASR) performance in tonal languages such as Mandarin, Thai, Vietnamese, etc. Since Myanmar language is being considered as a tonal language, the effect of tones on both syllable and word-based ASR performance has been explored. In this work, experiments are done based on the modeling of tones by integrating them into the phoneme set and incorporating them into the Convolutional Neural Network (CNN), state-of-the-art acoustic model. Moreover, to be more effective tone modeling, tonal questions are used to build the phonetic decision tree. With tone information, experiments show that compared with Deep Neural Network (DNN) baseline, the performance of CNN model achieves nearly 2% for word-based ASR or more than 2% for syllable-based ASR improvement over DNN model. As a result, the CNN model with tone information gets 2.43% word error rate (WER) or 2.26% syllable error rate (SER) reductions than without using it.

Keywords: Tone information

Automatic Speech Recognition (ASR) · Tonal language

Deep Neural Network (DNN) · Convolutional Neural Network (CNN)

1 Introduction

Nowadays, the performance of automatic speech recognition (ASR) systems is improved by exploring the new architecture and utilizing particular properties of the target language. Recently, Convolutional Neural Network (CNN) has shown significant performance in ASR tasks [1, 2]. CNN has gained higher performance than Deep Neural Network (DNN) across different large vocabulary continuous speech recognition (LVCSR) tasks [3, 4]. CNN has an ability to reduce the translational invariance and spectral correlations in the input signal. There are many

ASR tasks that utilize the particular features of the target language. For tonal languages such as Mandarin, Thai, Vietnamese, etc., they used the particular features of their language which means the tonal information was augmented to acoustic modeling to improve their ASR performance [5,6].

Myanmar language is assumed as a tonal language. There are four different tones in Myanmar language. It has different meanings according to the different types of tones. The following example shows the four tones of the phoneme ‘a’ (အ) and its different meanings in Myanmar language. /a te/ (အ တယ်) [to be wide open, to talk too much] with Tone1, /a: te/ (အ: တယ်) [to be free] with Tone2, /a. te/ (အ တယ်) [to be dumb or dull or naive] with Tone3 and /a' te/ (အံ တယ်) [to place an order] with Tone4 [8]. Therefore, accurate tone recognition plays an important role in automatic Myanmar speech recognition. In this work, the effect of tones is explored using state-of-the-art acoustic modeling approach, CNN-based acoustic model for Myanmar language. In low-resourced condition, CNN is better than DNN because the fully connected nature of the DNN can cause overfitting and it degrades the ASR performance for low-resourced languages where there is a limited amount of training data [1]. CNN can alleviate these problems and it is very useful for a low-resourced language such as Myanmar. Moreover, CNN can model well tone patterns because it can reduce spectral variations and model spectral correlations existing in the signal. For tonal languages such as Vietnamese [9], the tonal information are incorporated into the phonetic decision tree and it showed promising result. Therefore, in this work, tonal questions are used to build the phonetic decision tree in order to get more sophisticated tone modeling.

This paper is organized as follows: In Sect. 2, Myanmar language and previous Myanmar ASR research works are discussed. Building the speech and text corpus is presented in Sect. 3. About pronunciation lexicon is in Sect. 4. In Sect. 5, CNN is introduced. The experimental setup is performed in Sect. 6. The evaluation result is discussed in Sect. 7. Error analysis is done in Sect. 8. The conclusion and future work are presented in Sect. 9.

2 Myanmar Language and Previous Myanmar ASR Research Works

In this section, we describe Myanmar language and its previous ASR research works.

2.1 Nature of Myanmar Language

Myanmar language is a tonal, syllable-timed language, largely monosyllabic and analytic language with a subject-object-verb word order. There are 12 basic vowels, 33 consonants, and 4 medials in Myanmar language. The basic consonants in Myanmar can be extended by medials. Syllables or words are formed by consonants combining with vowels. Generally, one syllable is formed by combining

Table 1. Characteristics of Myanmar tones

Register		Phonation	Length	Pitch
Low	a	Modal voice	Medium	Low
High	a:	Breathy voice	Long	High
Creaky	a.	Creaky voice	Medium	High
Checked	a?	Final glottal stop	Short	Varies

Table 2. Three example Myanmar sentences of training data

<p>State Counsellor Daw Aung San Suu Kyi sends a message to the Myanmar Motion Picture Academy Awards Presentation Ceremony for 2016. နှစ် ထောင့် ဆယ် ရက်က နှစ် နှစ် နှစ် နှစ် အကယ်၍ ဝေပညာ အခမ်းအနား သို့ နိုင်ငံတော် စီ အတိုင်ပင်ခံ ပုဂ္ဂိုလ် ဒေါ် အောင်ဆန်းစုကြည် မှ သတိထား တစ် နောင် ပေးပို့ ခဲ့ ပါတယ် (nhi' ltaum. hse. chau' ku. nhi' mjan ma. jou' shin a- ke da- mi ltu: gyun lsu. pei: bwe: a- kha: a- na: dhou. nain gan do i. a- tain bin gan pou' gou do aun hsan: su. kyi mha. tha- wun lha ta- zau: pei: pou. ge. ba. de)</p> <p>The domestic gold price is an upward trend with the global price rising, according to the local gold market. ကမ္ဘာ့ ရွှေဈေး ပြင်ပက လာ မှ ရက်က ပြည်တွင်း ရွှေ ဈေးကွက် အမြင့် တက် သို့ နှိုင်းယှဉ် နေ ရက်က: ဒေသ ရွှေ ဈေးကွက် က ဆို ပါတယ် (ga- ba. shwei zet: mjin. te' la mhu. gyaun. pji dwin: shwei zet: gwe' a- mjin. be' thou. u: ti nei gyaun: del lha. shwei zet: gwe' ka. hsu ba de)</p> <p>The Lao Tennis Federation is selecting a squad to compete in the 29th Southeast Asian (SEA) Games to be held in Malaysia in August. လာအို တင်နစ် အဖွဲ့ချုပ် မှ တာမည် ပြိုင်ဘက် လာ မလေးရှား နိုင်ငံ တွင် ကျင်းပ မည့် နှစ် ဆယ် ကိုး ကြိမ်မြောက် အရှေ့တောင် အာရှ ဆိုက်စ် တွင် ပါဝင် ယှဉ်ပြိုင် ရန် အသင်း အတွက် အာကစားသမား များကို ရွေးချယ် နေ ပါတယ် (la ou tin: ni' a- hpwe. gyou' mha. la. me. o: gou' la. ma- lei: sha: nain ngan dwin kyin: pa. mji. nha- hse. kou: gyein mjaui' a- shei. tain a sha. hsi: gein: dwin pa win shin pjain jan a- thin: a- twe' a: ga- zac: tha- ma: mjin: gou jwei: che nei ba de)</p>

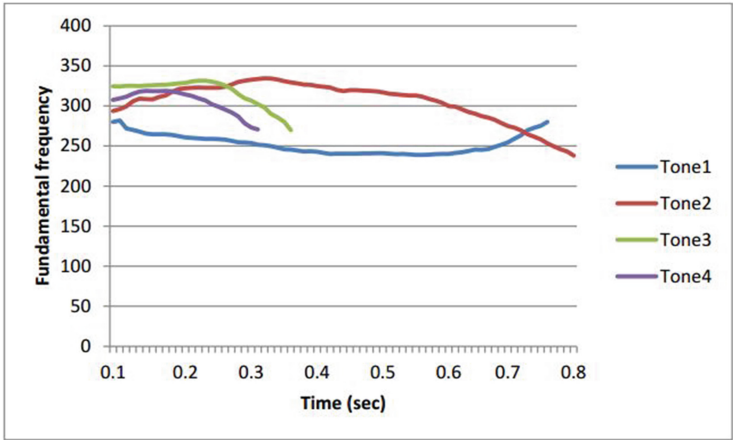


Fig. 1. Example of four tones of the Myanmar syllable ‘a’ (အ)

one consonant and vowel, and one word is composed of one or more than one syllable. There are 23 phonemes for 33 consonants since some consonant letters have the same phoneme. Myanmar tone is carried by syllable and is featured by both fundamental frequency and duration of syllable. There are four nominal tones transcribed in writing Myanmar: low, high, creaky and checked [7]. For example, the four types of tones of the phoneme ‘a’ (အ) are /a/ (အ), /a:/ (အ:), /a./ (အ.) and /a’/ (အ’)

and their fundamental frequency is shown in Fig. 1. Table 1 shows the characteristics of Myanmar tones.

2.2 Previous Myanmar ASR Research Works

This section discusses recent publications in Myanmar ASR. Hidden Markov Model with Subspace Gaussian Mixture Model (HMM-SGMM) continuous automatic speech recognition on Myanmar web news was developed by Mon et al. [10]. In this work, both Gaussian Mixture Model (GMM) and SGMM-based systems are compared using n-gram language model. Moreover, ASR performance was evaluated based on training data size, language model size and number of Gaussians. Soe and Thein [11] presented syllable-based speech recognition for Myanmar using GMM with Hidden Markov Model (GMM-HMM) approach. Syllable-based language model was used in this work. Myanmar language speech recognition with hybrid artificial neural network and hidden Markov model was demonstrated by Nwe and Myint [12]. This system used Artificial Neural Network and Hidden Markov Model (ANN-HMM) in acoustic model building and Mel Frequency Cepstral Coefficient (MFCC), Linear Predictive Cepstral Coding (LPCC) and Perceptual Linear Prediction (PLP) were used in feature extraction techniques. A Myanmar large vocabulary continuous speech recognition for travel domain was presented by Naing et al. [7]. In this work, DNN-based acoustic model was developed. Word error rate (WER) and syllable error rate (SER) were used as the evaluation criteria. The tonal phonemes and pitch features were applied in acoustic modeling and it showed that tones information is very helpful for Myanmar language. Myanmar continuous speech recognition based on Dynamic Time Wrapping (DTW) and HMM was expressed by Khaing [13] using HMM approach. Combinations of LPC, MFCC and Gammatone Cepstral Coefficients (GTCC) techniques were used in feature extraction and DTW was applied in feature clustering.

3 Building the Speech and Text Corpus

We already built the 5-hour-data set for the broadcast news domain. It was used in my first paper [10] and in that paper, we extend our 5-hour-data set to 10-hour-data set. To build the speech corpus, there are many websites on the Internet that the Myanmar broadcast news is available. Among them, we collected the recorded speech with clear voice from Myanmar Radio and Television (MRTV), Voice of America (VOA), Eleven broadcasting, 7 days TV, and ForInfo news. Both local news and foreign news are included in the corpus. The audio files are cut from each news file. And then, the segmented files are set to a single channel (mono type) and the sampling rate is set to 16 kHz. The length of the segmented audio files is between 2 s and 25 s. Most of the audio files from the corpus are female voice, and only few audio files are male voice. The average number of words and syllables in one utterance is 33 words and 54 syllables. Most of the broadcast news at the online never transcribe into text. Therefore, the audio data is listened to and transcribed into text manually. Moreover, Myanmar words do not have boundary in writing like the other languages such as English. In order to define the word boundary, it needs to put a space between Myanmar words and this is done by using Myanmar word segmenting tool [14]. The segmentation and

spelling of the words are then manually checked. To increase the size of the text corpus, bootstrapping technique is used. The total time taken for preparing the corpus is about 5 months. Table 2 shows the three example Myanmar sentences from the training set.

4 Pronunciation Lexicon

Lexicon is a list of words, with a pronunciation for each word expressed as a phone sequence. To generate the pronunciation of the new words, grapheme-to-phoneme (g2p) converter [15] is used and then the new words are added to the lexicon. In this experiment, two types of dictionary are used: dictionary with tone and dictionary without tone.

4.1 Tonal Pronunciation Lexicon

A standard dictionary, Myanmar language commission (MLC) dictionary with tone, is adopted as baseline [16] and this dictionary is extended to words from the speech corpus. There are about 36,700 words in the lexicon. Table 3 shows a snippet of Myanmar phonetic dictionary with tone.

Table 3. Example of Myanmar phonetic dictionary with tone

Myanmar word	Phonetic
က	k a.
ကာ	k a
ကစား	g a- z a:
ကသိ	k a- th i'

4.2 Non-tonal Pronunciation Lexicon

Tone information is not included in this dictionary and Table 4 depicts a snippet of Myanmar phonetic dictionary without tone information.

Table 4. Example of Myanmar phonetic dictionary without tone

Myanmar word	Phonetic
က	k a
ကာ	k a
ကစား	g a z a
ကသိ	k a th i

5 Convolution Neural Network (CNN)

CNN has shown a greater success in ASR tasks than Deep Neural Network (DNN) recently [3, 4] because of local filtering and max pooling layer of the CNN architecture where DNN has only the fully connected layers. Therefore, CNN has an advantage of handling in small frequency shifts that are common in speech signal and temporal variability due to varying speaking rate. CNN runs a small window over the input speech so that the weights of the network that looks through this window can learn from various features of the input.

Convolution layer - In this layer, each hidden activation h_i takes inputs from a small rectangular section of the previous layer, multiplying those local inputs (i.e., $[v_1, v_2, v_3]$) against the weight matrix W . The weight matrix, or the localized filter, will be replicated across the entire input space to detect a specific kind of local pattern. All neurons sharing the same weights compose a feature map. A complete convolutional layer is composed of many feature maps, generated with different localized filters, to extract multiple kinds of local patterns at every location.

Pooling layer - The pooling layer similarly takes inputs from a local region of the previous convolutional layer and down-samples it to produce a single output from that region. One common pooling operator used for CNNs is max-pooling, which outputs the maximum value within each sub-region.

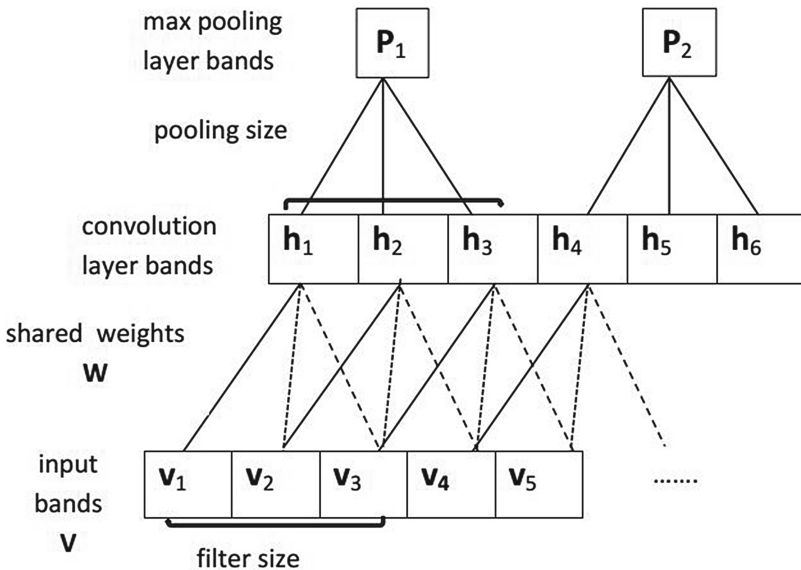


Fig. 2. A typical convolutional network architecture consisting of a convolutional and max-pooling layer. In this figure, weights with the same line style are shared across all convolutional layer bands with non-overlapping pooling.

One or more fully connected hidden layers are added on top of the final CNN layer in order to combine the features across all frequency bands. The Softmax layer interprets network output scores as conditional probabilities, for each class label. A typical convolutional neural network architecture [1] is shown in Fig. 2.

6 Experimental Setup

6.1 Data Sets

For training the acoustic model, a 10-hour-data set is used. There are 102 speakers with 3,530 utterances in the training set in total. For evaluating the performance, about 32-minute-test set is applied and there are 8 speakers (5 females and 3 males) with 193 utterances.

Table 5 shows the detailed information of the data sets.

Table 5. Train data and test data used in the experiment

Data	Size	Speakers			Utterance	UniqueWords
		Female	Male	Total		
TrainSet	10 h	79	23	102	3,530	2,590
TestSet	31 min 55 s	5	3	8	193	785

6.2 Word-Based and Syllable-Based ASR

Two ASR experiments are performed and they are word-based and syllable-based ASRs.

For the word-based ASR, training data and testing data are segmented by word units. The language model is trained from the word segmented training data and lexicon is also constructed at word level. The experiments show the ASR performance at the word level. WER is the typical ASR evaluation criterion and it is used to evaluate the word-based ASR performance.

In order to build the syllable-based ASR, all training data and testing data are segmented by the syllable units using the syllable breaking tool¹. And then, the language model is built from syllable segmented training data. Lexicon is prepared with syllable units. For the syllable-based ASR, another alternative evaluation criterion, SER, is used to evaluate the ASR performance based on syllable units.

6.3 Language Model

The language model (LM) is built by using the transcription of all training speech with SRILM [17] language modeling toolkit. For both word-based and syllable-based ASR, 3-gram language model is used.

¹ <https://github.com/ye-kyaw-thu/sylbreak>.

6.4 Features and Acoustic Model

All the experiments are performed using Kaldi [18] speech recognition toolkit. In the experiments, three types of acoustic model are built and they are baseline (GMM-HMM), DNN and CNN. The Kaldi pitch tracker [19] is used to extract tone related pitch features which are augmented features to acoustic model. In the features, there are 3-dimensional pitch features (normalized pitch, delta-pitch, and voicing features).

GMM-HMM - As input features, mel frequency cepstral coefficients (MFCC) with delta and delta features with Cepstral Mean and Variance Normalization (CMVN). And then, Linear Discriminative Analysis (LDA) and Maximum Likelihood Linear Transform (MLLT) are applied to 9-splce frames and project to a 40-dimension feature. Then, Speaker Adaptive Training (SAT) is performed on LDA+MLLT model. It has 2,095 context dependent (CD) triphone states and an average of 14 Gaussian components per state.

DNN - 40-dimensional log mel-filter bank features with per-speaker mean and variance normalization are used as input features. No pre-training is used. There are 4 layers with 1,024 units per hidden layers.

CNN - The input features are the same with DNN. There are two convolution layers, one pooling layer and two fully connected layers with 1,024 hidden units. In the experiment, 1-Dimensional convolutional network is used. The size of the pooling is 3 and max pooling is used. There are 128 feature maps in the first convolutional layer and 256 feature maps in the second convolutional layer. 8×3 filter size is used and no pre-training is applied for the CNN experiment.

6.5 Tones Clustering Using Phonetic Decision Trees

The basic idea of using decision tree is to find similar triphones, and share the parameters between them. One of the biggest advantages of the decision trees is that they do not limit the scope of the questions in any way. This makes it possible to incorporate different sources of information in the decision process [20]. In Kaldi, the questions used to build the decision tree are generated automatically based on the tree clustering of the phones. Because of the flexible structure of Kaldi framework, extra questions for linguistic knowledge are allowed to add further tuning for a particular language. Therefore, two simple set of questions about tones are added for Myanmar language. They are questions to group phonemes with the same base vowel together and questions to group phonemes with the same tone together. Table 6 shows two examples of the tonal extra questions to incorporate into the decision tree.

Table 6. Example of the tonal extra questions to incorporate into decision tree

Same base vowel	Same tone
a. a: a: a'	a. i. ei. an. e. in. o. ou. u. un.
ei. ei ei: ei'	a i ei an e in o ou u un

7 Evaluation Result

In this section, three experiments are used to evaluate the ASR performance.

Experiment1 (Exp1)

For the Experiment1, non-tonal dictionary, standard MFCC features for GMM model, and log Mel-filter bank (FBank) features for DNN and CNN are applied.

Experiment2 (Exp2)

The augmenting of pitch into acoustic features of each model and non-tonal dictionary are used in this setup.

Experiment3 (Exp3)

The last experiment, Experiment3 is with pitch features, tonal dictionary and the incorporation of tones into the phonetic decision tree.

Table 7. Evaluation of word-based ASR performance over tone and pitch features

Models	WER%		
	Exp1	Exp2	Exp3
GMM-HMM	46.08	44.92	42.33
DNN	45.04	42.14	39.89
CNN	40.39	39.69	37.96

In this experiment, the effect of pitch and tone features are explored for word-based ASR performance. Table 7 shows that the evaluation result over tone and pitch features. The lowest WER is shown in highlighted for each experiment. Without using tonal dictionary, Exp1 achieves WER about 46.08% for GMM model, 45.04% for DNN model and 40.39% for CNN model. As a result, CNN gains 5.69% and 4.65% absolute improvement over GMM and DNN. After augmenting the pitch features into each model, Exp2 gets WER reduction on 1.16%, 2.9% and 0.7% over GMM, DNN and CNN of Exp1. These results show that pitch features give better accuracy on all three models than without using it. Exp3 with pitch, tonal dictionary and the incorporation of tones to the decision tree achieves better results (3.75% and 2.59% absolute) on the conventional GMM model than Exp1 and Exp2. In comparison with Exp1 and Exp2, it notably improves the accuracy of 5.15% and 2.25% absolute in the DNN and 2.43% and 1.73% absolute in the CNN model. Using the tone and pitch features, it achieves the best WER of 37.96% with CNN.

Table 8 shows the syllable-based ASR performance over tones and pitch features. Exp1 without using tones and pitch information, it achieves SER about 34.96% on GMM, 34.63% on DNN, and 29.50% on CNN. For all models, it is observed that SER reduces greatly when the tones and pitch information is

Table 8. Evaluation of syllable-based ASR performance over tone and pitch features

Models	SER%		
	Exp1	Exp2	Exp3
GMM-HMM	34.96	33.98	31.44
DNN	34.63	32.79	29.47
CNN	29.50	27.84	27.24

used and the lowest SER is also highlighted. For the GMM-based model with Exp3, it achieves 3.52% and 2.54% absolute better than Exp1 and Exp2. Exp3 of DNN-based model gains better accuracy of 5.16% and 3.32% over the other two experiments. It shows that CNN-based model with tones and pitch information gets SER reduction of 2.26% and 0.6% than Exp1 and Exp2. Among the three different models, CNN is the best with 2.23% and 4.2% absolute better than DNN and GMM.

According to the Tables 7 and 8, it can be clearly seen that both tonal information and pitch features are important to improve ASR performance in Myanmar language.

8 Analysis of the Results of Tone Recognition

In this section, analysis results for four types of tones are discussed. The analysis is done to improve the future ASR performance. The tone error confusion matrices for both word and syllable-based ASR are shown in Tables 9 and 10. The highlighted values are the highest and the lowest accuracy of the tones. According to the Tables 9 and 10, it is observed that the overall accuracy, 88.71%, is achieved for word-based ASR and for syllable-based ASR, the average accuracy, 95.23%, is obtained. Tone1, which is the highest percentage of distribution among all tones, gets the best accuracy, 91.25% for the word-based model, and 97.25% for the syllable-based model. The lowest accuracy is got with Tone4, which is the least frequent tones. With respect to Tone4, there are about 87.28% accuracy in the word-based ASR and 93.90% in the syllable-based ASR.

Table 9. Confusion matrix of tone recognition for word-based ASR

	Tone1	Tone2	Tone3	Tone4	Accuracy %
Tone1	3,472	123	151	59	91.25
Tone2	141	1,786	82	36	87.33
Tone3	156	55	2,173	58	88.98
Tone4	40	43	78	1,105	87.28
Overall accuracy %					88.71

Table 10. Confusion matrix of tone recognition for syllable-based ASR

	Tone1	Tone2	Tone3	Tone4	Accuracy%
Tone1	5,683	63	72	26	97.25
Tone2	83	2,566	42	12	94.93
Tone3	107	53	3,550	34	94.82
Tone4	31	31	37	1,524	93.90
	Overall accuracy %				95.23

Among the tone confusion pairs, the pairs of Tone1 with Tone3, Tone3 with Tone1, and Tone2 with Tone1 are most of the confused tone pairs. This is due to the speaking styles of news presenters. For example, the tone confusion pairs of Tone1 with Tone3 are က (k a) is with က (k a.), တေ (t e) is with တေ (t e.), etc. Similarly, some of the example confusion pairs of Tone3 with Tone1 are အ (a.) is with အ (a), အိ (i.) is with အိ (i), etc. The confusion tone pairs of Tone2 with Tone1 are အိ (ou:) is with အိ (ou), အ (a:) is with အ (a), etc.

The analysis will be taken into account for the future research to improve the accuracy of all tones.

9 Conclusion

In this work, the effect of tone and pitch features was explored using state-of-the-art acoustic modeling technique, CNN, at both syllable and word levels of Myanmar ASR. It clearly shows that CNN notably outperforms DNN for tonal language like Myanmar. Moreover, the addition of tones into the phoneme set and acoustic features, and using the tonal extra questions into the building of phonetic decision tree are proved that they help to improve the ASR performance for Myanmar language. Using tone and pitch features, with GMM, DNN, and CNN-based models, better accuracy of 42.33%, 39.89% and 37.96% are achieved at the word level, and 31.44%, 29.47% and 27.24% are gained at the syllable level. As a result, in comparison with DNN, word-based CNN offers nearly 2% improvement and syllable-based CNN gains over 2% better accuracy with respect to the tone and pitch features. Furthermore, the CNN model with tone information gets WER of 2.43% or SER of 2.26% reductions than without using it.

For future work, more experiments will be done on CNN architecture using different parameters (number of feature maps, number of hidden units, number of convolutional layers, etc.) and pooling techniques. Moreover, more tonal features will be explored for Myanmar language.

References

1. Sainath, T.N., Mohamed, A., Kingsbury, B., Ramabhadran, B.: Deep convolutional neural networks for LVCSR. In: IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, pp. 8614–8618, 26–31 May 2013
2. Sainath, T.N., Kingsbury, B., Mohamed, A., Dahl, G.E., Saon, G., Soltau, H., Beran, T., Aravkin, A.Y., Ramabhadran, B.: Improvements to deep convolutional neural networks for LVCSR. In: 2013 IEEE Workshop on Automatic Speech Recognition and Understanding, Olomouc, pp. 315–320, 8–12 December 2013
3. Sainath, T.N., Kingsbury, B., Saon, G., Soltau, H., Mohamed, A., Dahl, G.E., Ramabhadran, B.: Deep convolutional neural networks for large-scale speech tasks. *Neural Netw.* **64**, 39–48 (2015)
4. Sercu, T., Puhrsch, C., Kingsbury, B., LeCun, Y.: Very deep multilingual convolutional neural networks for LVCSR. In: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2016, Shanghai, pp. 4955–4959, 20–25 March 2016
5. Hu, X., Saiko, M., Hori, C.: Incorporating tone features to convolutional neural network to improve Mandarin/Thai speech recognition. In: Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, APSIPA 2014, Chiang Mai, pp. 1–5, 9–12 December 2014
6. Nguyen, V.H., Luong, C.M., Vu, T.T.: Tonal phoneme based model for Vietnamese LVCSR. In: 2015 International Conference Oriental COCODA Held Jointly with 2015 Conference on Asian Spoken Language Research and Evaluation (O-COCOSDA/CASLRE), Shanghai, pp. 118–122, 28–30 October 2015
7. Naing, H.M.S., Hlaing, A.M., Pa, W.P., Hu, X., Thu, Y.K., Hori, C., Kawai, H.: A Myanmar large vocabulary continuous speech recognition system. In: Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, (APSIPA 2015), Hong Kong, pp. 320–327, 16–19 December 2015
8. Htun, U.T.: Acoustic Phonetics and the Phonology of the Myanmar Language. School of Human Communication Sciences, La Trobe University, Melbourne (2007)
9. Luong, H.T., Vu, H.Q.: A non-expert Kaldi recipe for Vietnamese speech recognition system. In: Proceedings of WLSI/OIAF4HLT, Osaka, pp. 51–55, 12 December 2016
10. Mon, A.N., Pa, W.P., Thu, Y.K.: Building HMM-SGMM continuous automatic speech recognition on Myanmar web news. In: Proceedings of 15th International Conference on Computer Applications (ICCA 2017), pp. 446–453 (2017)
11. Soe, W., Thein, Y.: Syllable-based speech recognition system for Myanmar. *Int. J. Comput. Sci. Eng. Inf. Technol. (IJCEIT)* 1–13 (2015)
12. Nwe, T.T., Myint, T.: Myanmar language speech recognition with hybrid artificial neural network and Hidden Markov Model. In: Proceedings of 2015 International Conference on Future Computational Technologies (ICFCT 2015), Singapore, pp. 116–122, 29–30 March 2015
13. Khaing, I.: Myanmar continuous speech recognition system based on DTW and HMM. *Int. J. Innov. Eng. Technol. (IJIET)* **2**(1), 78–83 (2013). ISSN 2319-1058
14. Pa, W.P., Thu, Y.K., Finch, A., Sumita, E.: Word boundary identification for Myanmar text using conditional random fields. In: Zin, T.T., Lin, J.C.-W., Pan, J.-S., Tin, P., Yokota, M. (eds.) GEC 2015. AISC, vol. 388, pp. 447–456. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-23207-2_46

15. Thu, Y.K., Pa, W.P., Finch, A., Ni, J., Sumita, E., Hori, C.: The application of phrase based statistical machine translation techniques to Myanmar grapheme to phoneme conversion. In: Computational Linguistics - 14th International Conference of the Pacific Association for Computational Linguistics, PACLING 2015, Bali, Revised Selected Papers, pp. 238–250, 19–21 May 2015
16. Myanmar-English Dictionary: Department of the Myanmar Language Commission. Yangon, Ministry of Education, Myanmar (1993)
17. Stolcke, A.: SRILM - An Extensible Language Modeling Toolkit, pp. 901–904 (2002)
18. Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., Hannemann, M., Motlicek, P., Qian, Y., Schwarz, P., Silovsky, J., Stemmer, G., Vesely, K.: The Kaldi speech recognition toolkit. In: IEEE 2011 Workshop on Automatic Speech Recognition and Understanding. IEEE Signal Processing Society. IEEE Catalog No.: CFP11SRW-USB, December 2011
19. Ghahremani, P., BabaAli, B., Povey, D., Riedhammer, K., Trmal, J., Khudanpur, S.: A pitch extraction algorithm tuned for automatic speech recognition. In: IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2014, Florence, pp. 2494–2498, 4–9 May 2014
20. Hirsimäki, T.: A review: decision trees in speech recognition, 27 May 2003

Spoken Language and Dialogue



Listenability Measurement Based on Learners' Transcription Performance

Katsunori Kotani^{1(✉)} and Takehiko Yoshimi²

¹ Kansai Gaidai University, Osaka, Japan
kkotani@kansaigaidai.ac.jp

² Ryukoku University, Shiga, Japan

Abstract. Language teachers using listening materials on the Internet need to examine the ease of the listening materials (hereafter, listenability) they choose in order to maintain learners' motivation for listening practice since the listenability of such materials is not controlled, unlike commercially available teaching materials. This study proposes to use a listenability index based on learners' transcription performance. Transcription performance was determined using the normalized edit distance (hereafter, NED) from a learner's transcription to a reference sentence. We examined the reliability and validity of NED as a dependent variable for listenability measurement using multiple regression analysis in an experiment comprising 50 learners of English as a foreign language. The results supported the reliability and validity of NED.

Keywords: Listenability · English as a foreign language · Transcription performance · Edit distance

1 Introduction

A teaching assistance method that measures the ease of listening materials (hereafter, listenability) for learners of English as a foreign language has been reported to provide an educational application using natural language techniques [1–3]. Listenability should be controlled to match a learner's proficiency to maintain their motivation to practice listening skills [4]. Listenability is well-controlled in commercially available teaching materials, but not in the listening materials currently available on the Internet. Hence, language teachers must examine listening materials and choose those with listenability appropriate for their learners. This task is both time and effort consuming. The use of listenability measuring methods decreases the need for teachers to examine listenability, which leads to increased teaching efficiency. In addition, since listenability measuring methods can be implemented on a computer-assisted language learning system, learners can choose appropriate online listening materials on their own.

Under previous methods [1–3], listenability was measured by multiple regression analysis using linguistic and learner features as independent variables, and learners' subjective judgment for listening (hereafter, SBJ) as a dependent variable. Learners subjectively judged overall listening comprehension, and determined listenability on a five-point Likert scale as follows: (i) sentence understanding from easy to difficult

[1, 2], (ii) overall understanding from misunderstanding of the main topics to perfect understanding; (iii) understanding rate from less than 60% to 100%; (iv) recall rate from less than 60% to 100%; (v) the number of missed words from more than 10 words to none; and (vi) speech rate from fast to slow [3]. The validity of these dependent variables was confirmed based on statistically significant results of listenability measurement.

This study explores the possibility of using transcription performance as another dependent variable for measuring listenability. The primary advantage of transcription performance is its objective validity of evaluation criteria for language tests [5]. Other advantages are the possibility that transcription performance demonstrates implicit listening difficulty that learners fail to notice subjectively, as well as the broadened scope of listenability measurement. Transcription performance can be examined at the word level; hence, listenability is measurable on a word-by-word basis. However, SBJ has only previously been measured at the sentence level [1, 2] or text level [3].

Given these advantages, this study assumes that transcription performance can be determined with the normalized edit distance (hereafter, NED) from a sentence transcribed by a learner to a reference spoken sentence, and addresses the following research questions:

- How stable is NED as an evaluation criterion?
- To what extent does NED classify learners depending on their proficiency?
- How strongly does NED correlate with a learner's proficiency?
- How accurately is NED measurable based on linguistic and learner features for listenability measurement?

Note that NED refers to the normalized Levenshtein edit distance, which is calculated by dividing the Levenshtein edit distance by the number of characters in a longer sentence than the other. The Levenshtein edit distance reflects the differences between two sentences due to substitution, deletion, or insertion of characters.

The first three questions were answered within the framework of the classical test theory [6], and NED was confirmed as being a reliable and valid dependent variable for listenability measurement. The final question was answered by multiple regression analysis, which demonstrated a strong correlation ($r = 0.87$) between observed and measured values in a leave-one-out cross validation test.

2 Learner and Linguistic Features

Learner features represent effective independent variables for listenability measurement by explaining a learner's proficiency, according to Kotani et al. [1, 2]. In this study, learner proficiency was determined using English language test scores. Among the various English tests, this study employed the Test of English for International Communication (TOEIC), which is a major English language test for university learners in the country where this study was conducted.

Linguistic features explaining linguistic complexity also represent effective independent variables for listenability measurement [1–3, 7–11]. The linguistic features used

in this study were sentence length, word length, multiple syllable words, word difficulty, and speech rate.

These linguistic features were automatically derived as follows. Sentence length was derived as the number of words in a sentence. Word length was derived as the mean number of syllables in a word in a sentence. Multiple syllable words were derived as $\sum_{i=1}^N (S_i - 1)$, where N was the number of words in a sentence, and S_i was the number of syllables in the i -th word [8]. This subtraction derivation eliminated the presence of single-syllable words. Word difficulty was derived as the rate of words not listed in a basic vocabulary list [9] relative to the total number of words in a sentence. Speech rate was derived as the number of words read aloud in 1 min.

3 Compilation of Listenability Data

This study compiled two types of dependent variables for listenability measurement: NED and SBJ. NED was calculated between a learner's transcription and a reference sentence that a learner heard and transcribed. SBJ was derived as the score that a learner subjectively determined on a five-point Likert scale for the listenability of a sentence (1: easy, 2: somewhat easy, 3: average, 4: somewhat difficult, 5: difficult).

These dependent variables were compiled from 50 learners of English as a foreign language at university (28 males, 22 females; mean age: 20.8 years old [1.3 (standard deviation, SD)] who were compensated for their participation. All learners were asked to submit valid scores from TOEIC tests taken in the current or previous year. TOEIC scores range from 10–990. In our sample, the mean TOEIC score was 607.7 (186.2). The minimum score was 295 and the maximum score was 900.

Listening materials in this experiment were produced based on two texts distributed by the International Phonetic Association [11]. It was confirmed that each text included all of the English phonemes [11]. As these texts had no speech sound data, speech sounds were recorded from a voice actor's reading the texts aloud. The voice actor (female, 35 years old) read the texts aloud in an American accent. Table 1 summarizes the titles of the texts, text length, and the mean (SD) values of sentence length, word length, multiple syllable words, word difficulty, and speech rate. Note that the speech rate was slower than the natural speech rate (approximately 250 syllables per minute) [12].

Table 1. Characteristics of the listening materials.

Title	The north wind and the sun	The boy who cried wolf
Text length (sentences)	5	10
Text length (words)	113	216
Sentence length (words)	22.6 (8.3)	21.6 (7.6)
Word length (syllables)	1.3 (0.1)	1.2 (0.1)
Multiple syllable words (syllables)	6.4 (2.8)	5.7 (3.0)
Word difficulty	0.3 (0.1)	0.2 (0.1)
Speech rate (words per minute)	175.3 (12.8)	180.0 (19.8)

Learners listened to the listening materials sentence-by-sentence twice using headphones. After listening to a sentence, learners determined the SBJ for listenability on a five-point scale. Their judgments were recorded on home-built transcription tools. Learners then wrote down what they heard using the home-built transcription tools. While listening, they were allowed to take notes for transcription.

4 Properties of Listenability Data

The listenability data were compiled using the method described in Sect. 3. The descriptive statistics for the listenability data are summarized in Table 2.

Table 2. Descriptive statistics for SBJ and NED.

	SBJ	NED
Minimum	1	0.0
Maximum	5	1.0
Mean	4.2	0.5
<i>SD</i>	0.8	0.2
<i>N</i>	750	750

As listenability in terms of SBJ was classified into five levels, the NED was also classified into five levels to compare the data distributions in relative cumulative frequency, as seen in Fig. 1. The distributions were dissimilar, as most of the NED (approximately 90%) appeared between the listenability levels 1 and 3 (“easy”), while most of the SBJ (approximately 90%) appeared between the listenability levels 3 and 5 (“difficult”). That is, a listening material was judged more difficult with SBJ than NED. This suggests that learners listen with too much caution.

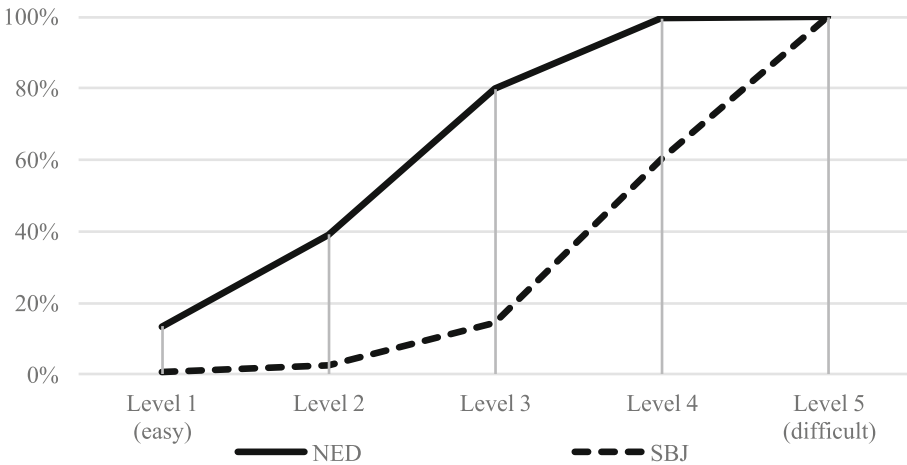


Fig. 1. Distribution of listenability.

5 Assessment of NED-Based Listenability

This study assessed NED as a dependent variable for listenability measurement based on the classical test theory [6]. Under the classical test theory, the reliability and validity of NED were assessed statistically.

The first research question was addressed by reliability, as described in Sect. 5.1. The second question was addressed by construct validity, as described in Sect. 5.2. The third question was addressed by criterion-related validity, as described in Sect. 5.2.

5.1 Reliability of NED-Based Listenability

The reliability of NED was examined by internal consistency in terms of Cronbach's α [13]. Internal consistency refers to whether NED demonstrates similar results for sentences with similar listenability. Cronbach's α is a reliability coefficient defined in Eq. (1), where α is the reliability coefficient, k is the number of items, in this case sentences, S_i^2 is the variance associated with item i , and S_T^2 is the variance associated with the sum of all k -item values. Cronbach's α reliability coefficient ranges from 0 (absence of reliability) to 1 (absolute reliability), and empirical satisfaction is achieved with values above 0.8. Note that the reliability coefficient of SBJ was also derived as reference data.

$$\alpha = \frac{k}{k-1} \left(1 - \frac{\sum_{i=1}^k S_i^2}{S_T^2} \right) \quad (1)$$

Since the reliability depends on the number of items, the reliability coefficients were derived in the North Wind and the Sun, which were composed of five sentences, and the Boy who Cried Wolf composing 10 sentences, and both texts containing 15 sentences. The NED and SBJ reliability coefficients are summarized in Table 3.

Table 3. Cronbach's alpha coefficients.

	NED	SBJ
The North Wind and the Sun	0.94	0.86
The Boy who Cried Wolf	0.96	0.85
Both texts	0.97	0.89

Both NED and SBJ exceeded the baseline reliability coefficient (0.8), and NED demonstrated higher reliability than SBJ. These results suggest that NED is more reliable than SBJ as a dependent variable for measuring listenability.

5.2 Validity of NED-Based Listenability

The validity of NED was examined by construct validity and criterion-related validity. The validity of NED is confirmed if it reflects learner proficiency-dependent listenability. The dependence of listenability on a learners' proficiency refers to instances in

which learners at different proficiency levels listen to a sentence and the listenability is higher for learners at a high proficiency level than for those at a low proficiency level.

Construct Validity. Construct validity was examined from the viewpoint of distinctiveness. If NED properly reflects a learner’s proficiency, NED should demonstrate a statistically significant difference ($p < 0.01$) among learners at different proficiency levels. The distinctiveness of NED was investigated using analysis of variance (ANOVA). Note that construct validity was also examined for SBJ as reference data.

First, 50 learners were classified into three levels based on their TOEIC scores, which ranged from 295 to 450 at the beginner level, from 490 to 685 at the intermediate level, and from 730 to 900 at the advanced level. Then, the mean NED and SBJ were calculated for each learner. Table 4 shows the mean (*SD*) values of the TOEIC scores, NED, and SBJ between the three groups.

Table 4. TOEIC scores and listenability.

	Beginner ($n = 240$)	Intermediate ($n = 240$)	Advanced ($n = 270$)
TOEIC	394.1 (54.0)	583.4 (56.7)	819.2 (56.1)
NED	0.6 (0.2)	0.5 (0.2)	0.3 (0.2)
SBJ	4.5 (0.7)	4.3 (0.7)	3.9 (0.9)

One-way ANOVA showed statistically significant differences ($p < 0.01$) between the three groups of learners for TOEIC scores, NED, and SBJ. The results were further examined using Tukey’s post-hoc comparison.

Tukey’s post hoc test showed statistically significant differences ($p < 0.01$) between the beginner and intermediate groups, the beginner and advanced groups, and the intermediate and advanced groups for TOEIC and NED, but not between the beginner and intermediate groups for SBJ. These results suggest that NED is more valid than SBJ as a dependent variable for listenability measurement. The results are summarized in Table 5.

Table 5. ANOVA and post-hoc test results.

	ANOVA		Post-hoc	
	<i>F</i> -value (<i>df</i>)	<i>P</i> -value	Pair	<i>P</i> -value
TOEIC	3741.1 (2, 74)	< 0.01	Beg.-Int.	<0.01
			Beg.-Adv.	<0.01
			Int.-Adv.	<0.01
NED	217.1 (2, 747)	< 0.01	Beg.-Int.	<0.01
			Beg.-Adv.	<0.01
			Int.-Adv.	<0.01
SBJ	38.3 (2, 7474)	< 0.01	Beg.-Int.	n.s.
			Beg.-Adv.	<0.01
			Int.-Adv.	<0.01

Criterion-Related Validity. Criterion-related validity was examined from the viewpoint of the correlation with learners' proficiency in terms of TOEIC scores. NED should reflect learners' proficiency because listenability should depend on learners' proficiency.

Although NED and SBJ were assigned to the sentences, TOEIC scores were assigned to the learners. Then, the mean NED and SBJ values were calculated for learners by dividing the sum of NED/SBJ values by the number of sentences (15 sentences). According to a technical TOEIC manual [14], empirically valid correlation coefficients were seen above 0.73, as TOEIC was correlated with a valid English test ($r = 0.73$).

TOEIC scores showed stronger correlations with NED ($r = -0.84$) than with SBJ ($r = -0.51$). The correlation coefficient for NED exceeded the baseline, but that of SBJ fell short of the baseline. NED and SBJ were further examined in an asymptotic z -test by using Fisher's z -transformation [15]. Statistically significant differences were observed ($n = 50$, $z = -3.2$, $p < 0.01$). The NED demonstrated more validity than the SBJ. The NED also demonstrated higher validity than the SBJ. These results provide further evidence for the validity of NED as a dependent variable for listenability measurement.

6 Listenability Measurement

This study developed a listenability measuring method using NED as a dependent variable in multiple regression analysis. In this section, the fourth research question is answered.

The independent variables were the linguistic and learner features described in Sect. 2. Before multiple regression analysis, the independent variables were examined in terms of the presence of multiple-collinearity by calculating the variance inflation factor (VIF) [16]. Multiple-collinearity ($VIF > 10$) was observed in multiple syllable words ($VIF = 12.3$); hence, multiple syllable words were excluded from the independent variables.

A significant regression equation was found ($F(5, 744) = 212.9$, $p < 0.01$), with an adjusted squared correlation coefficient (R^2) of 0.59, which means the equation succeeded in measuring approximately 59% of the listenability. The standardized partial regression coefficients of the linguistic and learner features are summarized in Table 6. The statistically significant independent variables were sentence length, word length, word difficulty, and TOEIC score, but not speech rate.

Table 6. Regression coefficients.

Feature	Standardized partial regression coefficient
Sentence length	0.40*
Word length	0.15*
Word difficulty	0.07*
Speech rate	-0.03
TOEIC score	-0.63*

* $p = 0.01$.

The listenability measurement method was also examined in a leave-one-out cross validation test in which the method was examined n times ($n = 750$), taking one instance as test data and $n - 1$ instances as training data. The correlation analysis was carried out between the observed and measured listenability values, and the adjusted R^2 was 0.58. The adjusted R^2 was then compared with that of the listenability measurement method using SBJ. The leave-one-out cross validation test results demonstrated that the adjusted R^2 using NED was higher than that using SBJ ($R^2 = 0.20$).

The listenability measurement method was also examined based on measurement errors in the cross-validation test. Errors were calculated as the absolute value of the difference between the observed and measured values. The distribution of errors is plotted in Fig. 2. Most of the errors were observed in the low error range ($0.0 < x \leq 0.1$).

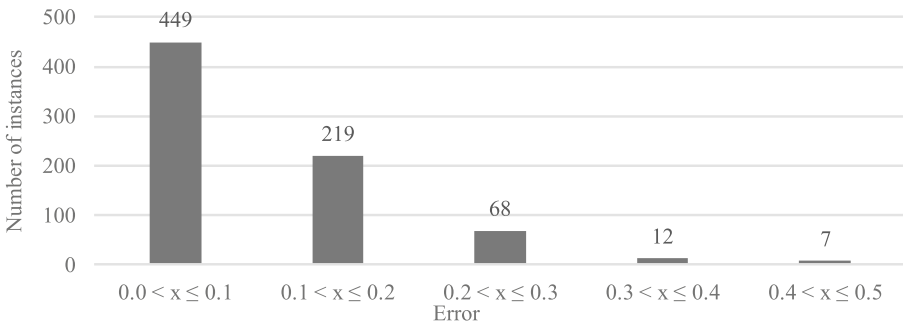


Fig. 2. Distribution of measurement errors.

Measurement errors were investigated to clarify what caused the errors. Independent sample t -tests were conducted to compare the linguistic and learner features between the top and bottom 10% of the errors (75 instances in each set). Table 7 shows the mean values and SD s. A statistically significant difference was observed only in sentence length ($t(148) = 3.3, p < 0.01$). This result suggests that the sentence length influenced the errors.

Table 7. Linguistic and learner features between the top and bottom 50 errors.

	Bottom 50 errors	Top 50 errors
Sentence length	18.6 (7.5)	22.4 (6.3)
Word length	1.3 (0.1)	1.3 (0.1)
Word difficulty	0.3 (0.1)	0.3 (0.1)
TOEIC score	620.0 (161.1)	605.1 (204.3)
Speech rate	176.3 (17.5)	177.7 (14.8)

7 Conclusion

This study proposed using NED to examine whether it properly demonstrated learners’ transcription performance as a dependent variable for listenability measurement. The

reliability and validity of NED were assessed using the classical test theory. The results suggest that NED is a reliable and valid dependent variable for listenability measurement. In addition, the use of sentence length needs some care. The contribution of sentence length to listenability measurement was confirmed as a statistically significant standardized partial regression coefficient as shown in Table 6. On the other hand, among the independent sentence length was a sole dependent variable that increased measurement errors, as seen in Table 7.

Acknowledgment. The authors would like to thank the anonymous reviewers for their valuable comments and suggestions to improve the quality of the paper. All remaining inadequacies are ours alone. This work was supported by JSPS KAKENHI Grant Numbers, 22300299, 15H02940.

References

1. Kotani, K., Ueda, S., Yoshimi, T., Nanjo, H.: A listenability measuring method for an adaptive computer-assisted language learning and teaching system. In: Proceedings of the 28th Pacific Asia Conference on Language, Information, and Computation, pp. 387–394 (2014)
2. Kotani, K., Yoshimi, T.: Learner feature variation in measuring the listenability for learners of English as a foreign language. In: Proceedings of the 1st International Workshop on Emerging Technologies for Language Learning (2016)
3. Yoon, S.-Y., Cho, Y., Napolitano, D.: Spoken text difficulty estimation using linguistic features. In: Proceedings of the 11th Workshop on Innovative Use of NLP for Building Educational Applications, pp. 267–276 (2016)
4. Petrides, J.R.: Attitudes and motivation and their impact on the performance of young English as a foreign language learners. *J. Lang. Learn.* **5**(1), 1–20 (2006)
5. Buck, G.: *Assessing Listening*. Cambridge University Press, Cambridge (2001)
6. Brown, D.J.: *Testing in Language Programs*. Prentice Hall Regents, New Jersey (1996)
7. Chall, J.S., Dial, H.E.: Predicting listener understanding and interest in newscasts. *Educ. Res. Bull.* **27**(6), 141–153 + 168 (1948)
8. Fang, I.E.: The easy listening formula. *J. Broadcast. Electron. Media* **11**(1), 63–68 (1966)
9. Kiyokawa, J.: A formula for predicting listenability: the listenability of English language materials 2. *Wayo Women's Univ. Lang. Lit.* **24**, 57–74 (1990)
10. Messerklinger, J.: Listenability. *Center Engl. Lang. Educ. J.* **14**, 56–70 (2006)
11. International Phonetic Association: *Handbook of the International Phonetic Association: A Guide to the Use of the International Phonetic Alphabet*. Cambridge University Press, Cambridge (1999)
12. Robb, M.P., Gillon, G.T.: Speech rates of New Zealand English- and American English-speaking children. *Adv. Speech Lang. Pathol.* **9**(2), 1–8 (2007)
13. Cronbach, L.J.: *Essentials of Psychological Testing*. Harper & Row, New York (1970)
14. Chauncey Group International: *TOEIC Technical Manual*. Chauncey Group International, New Jersey (1998)
15. Lee, I.A., Preacher, K.J.: Calculation for the Test of the Difference Between Two Dependent Correlations with One Variable in Common (2013). <http://quantpsy.org>
16. Neter, J., Kutner, M., Wasserman, W., Nachtsheim, C.: *Applied Linear Statistical Models*, 4th edn. McGrawHill/Irwin, New York (1996)

Speech Pathology



A Robust Algorithm for Pathological-Speech Correction

Naim Terbeh^(✉) and Mounir Zrigui

LaTICE Laboratory, Faculty of Sciences of Monastir, 5000 Monastir, Tunisia
naim.terbeh@gmail.com, terbehnaim1987@gmail.com,
mounir.zrigui@fsm.rnu.tn

Abstract. The current work presents an original approach based on the probabilistic-phonetic modeling to develop an algorithm permitting the correction of pathological Arabic speech. For this purpose, we follow three steps. The first consists in detecting the voice defects manifesting in the Arabic speech. Second, the sounds begetting degraded speeches are identified. The last task consists in proposing an original algorithm based on probabilistic-phonetic modeling to correct the pathological pronunciations. The developed algorithm is highly efficient. Indeed, we have attained a correction performance of 97%. Accordingly, researchers in computer sciences, in speech therapy and in biology can support in our contribution to the pathological speeches processing.

Keywords: Arabic language · Healthy/pathological speech
Probabilistic-phonetic modeling · Vocal pathology · Speech correction

1 Introduction and State of the Art

Human-machine communication is in full progress, thus facilitating the accessibility to information and its treatments by introducing new faster methods to access information like voice commands. Nonetheless, vocal pathologies prevent several users from benefiting from this technology. In order not to exclude this population from voice communication with machines, several studies addressing the numerical accessibility have been established. In such work, researchers have tried to develop new platforms whose goal has been to rectify mispronunciations contained in a speech signal. We can mention:

- Vocal pathologies detection
 - A new solution for extracting robust and accurate characteristics to detect and classify the anomalies of voice that we find within the Arabic speech was suggested by Alsulaiman et al. [14]. They got a high performance on the basis of an autocorrelation and entropy within various frequency bands.
 - A new method for the detection of voice disorders in the Arabic speech was invented in [1] by Terbeh et al. they focused on the probabilistic-phonetic modeling.
 - A new algorithm for the detection of voice handicaps and for the classification of acoustic signals as healthy or pathological was recommended by Majidnezhad

et al. in [5]. To achieve that aim, the authors opted for the artificial neural networks [7].

- A new solution for the detection of degraded frames from speech signals was invented by Majidnezhad et al. in [6]. To attain that target, the authors utilized the Linde-Buzo-Gray algorithm [3] and the hidden Markov models [4].
- Speech correction:
 - A new methodology for the correction of faulty pronunciations due to bad articulation or insufficient air pressure was created in [12] by Ajibola et al. Firstly, the principal idea was to suppress the frames having actual defects. Secondly, suppressed pieces were reconstructed. Finally, the algorithm for recognizing the reconstructed speech was launched by the authors. On the basis of recognition results, the authors distinguished two cases:
 - In the case of recognizing a resulting speech, this rephrasing would be saved.
 - Otherwise, reconstructing the resulting speech could be erroneous.
 - On the basis of speech recognition system, a novel approach was suggested in [13] by Bassil et al. an error correction method for a post-editing automatic speech recognition error correction method as well as an algorithm that was in fact based on the online spelling was proposed in this work.

In spite of the richness of the literature with studies addressing pathological speech processing, the pathological spoken Arabic language has not been treated yet. Our contribution consists, for each speech sequence classified as pathological, in introducing a new methodology whose target is to correct pronunciation defects.

This paper is organized as follows. The invented model is detailed in Sect. 2. The details about the experiments are described in Sect. 3. The concluding remarks and future work are mentioned in Sect. 4.

2 Contribution

The goal of this paper is to localize and to correct mispronounced sounds contained in Arabic discourses. The main idea consists in comparing, between the model representing a referenced Arabic healthy pronunciation and the model summarizing the pronunciation appropriate to speakers who suffer from voice anomalies. The invented method can be detailed in three fundamental steps:

- Detecting the voice anomaly manifesting in the Arabic discourses
- Identifying the problematic sounds
- Correcting the detected mispronunciations

Figure 1 describes the operation of the proposed system:

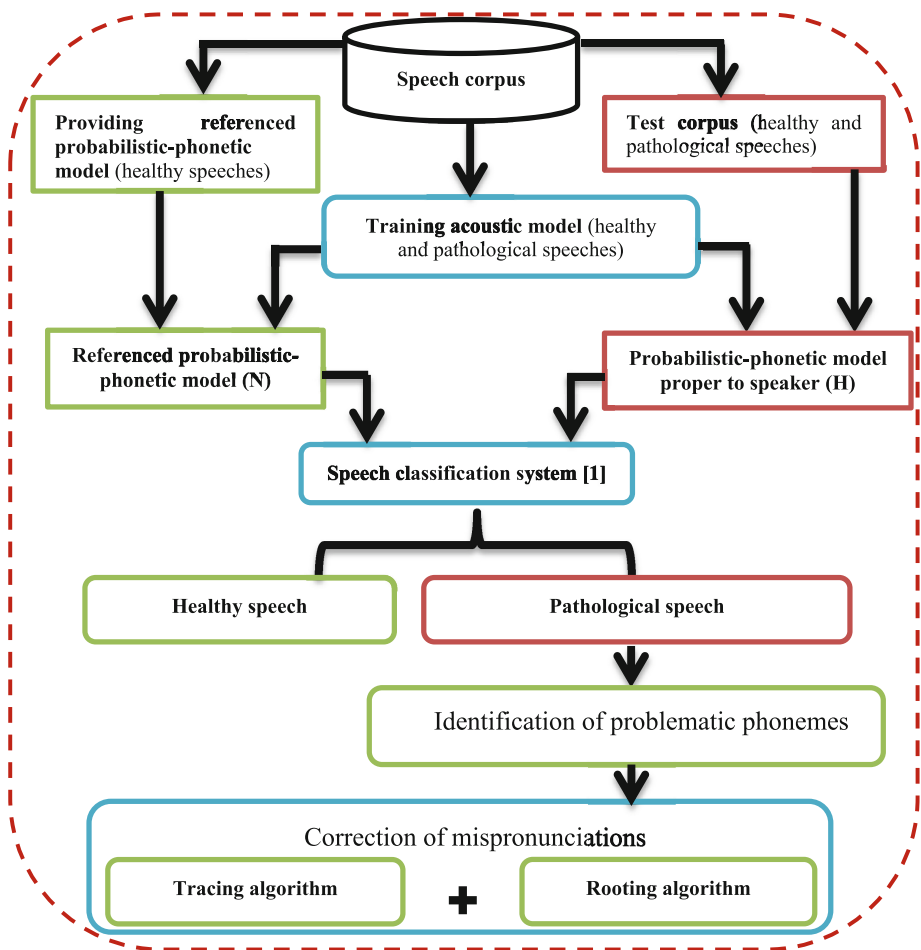


Fig. 1. General form of proposed system

2.1 Probabilistic-Phonetic Modeling and Speech Classification

To generate a referenced phonetic model referring to the Arabic vocabulary, we need an Arabic acoustic model and a large base of Arabic speech recorded by native and healthy speakers. After that, the Sphinx_align tool will be combined with the trained acoustic model to generate the phonetic transcription of our speech corpus. The probabilistic-phonetic model is defined by the vector summarizing the probability occurrences of each Arabic bi-phoneme, as shown in Fig. 2.

H_D	0.0015423348%	0.0015423348 = حد
H_DH	0.0046362397%	0.0046362397 = حد
H_R	9.346364E - 4%	9.346364E-4 = هر
H_Z	1.4776867E - 5%	1.4776861E-5 = هز
H_S	0.0%	0.0 = حس
H_SH	6.8342975E - 5%	6.8342975E-5 = هش
H_SS	0.0%	0.0 = حص
H_DD	3.140083E - 5%	3.140083E-5 = هض
H_TT	5.5413225E - 6%	5.5413225E-6 = هظ
H_DH2	0.0%	0.0 = حظ
H_AI	0.0%	0.0 = هع
H_GH	0.0%	0.0 = هغ
H_F	3.6942151E - 6%	3.6942151E-6 = هف
H_Q	3.8789258E - 5%	3.8789258E-5 = هق
H_K	1.4222728E - 4%	1.4222728E-4 = هك

Fig. 2. (a) Probabilistic-phonetic model referring to Arabic vocabulary: extract from obtained model using Arabic phonemes; and (b) extract from obtained model using Arabic letters

Next, we calculate the standard angle (phonetic distance) that separates between two different probabilistic-phonetic models referring to Arabic healthy speeches. For this task, we follow Algorithm 1:

Algorithm 1. Calculating referenced phonetic distance

1. We prepare n healthy speech corpus ($C_i, 1 \leq i \leq n$), and from each one, we determinate the probabilistic-phonetic model $M_i, (1 \leq i \leq n)$.
2. $S = \{\alpha_{ij}; 1 \leq i, j \leq n \text{ and } i \neq j\}$ a set of angles that separate between M_i and $M_j (\alpha_{ij} = \alpha_{ji} \text{ and } \alpha_{ii} = 0)$.
3. $Max = \text{maximum}\{S\}$.
4. $\delta = \text{standard deviation}\{S\}$.
5. $Avg = \text{average}\{S\}$ /*Avg defines the standard probabilistic-phonetic model referring the Arabic speech*/
6. $\beta = Max + |Avg - \delta|$ /*is the standard phonetic distance*/

For every new speaker, we calculate the angle θ that separates their proper probabilistic-phonetic model and that referenced to the Arabic speech (Avg). We distinguish two cases:

- If $\theta \leq \beta$, then the input speech is heathy.
- Else, ($\theta > \beta$) the input speech is pathological.

2.2 Problematic Sounds

This task of identifying problematic sounds will be applied if the input speech sequence is classified as pathological. We distinguish two main categories:

2.2.1 Substituted Sounds

Substituted phonemes pose mispronunciations for the concerned speakers. The main idea to identify these phonemes considers that a mispronounced phoneme does never appear in a speech phonetic transcription. Consequently, we have a coefficient equal to zero in the phonetic model for all bi-phonemes containing a wrongly pronounced phoneme.

A simple comparison between N and H models can lead to the identification of the phonemes that pose mispronunciations. Algorithm 2 is used to solve this problem:

Algorithm 2. Detecting substituted sounds

1. We propose these two sets M and G with $M=G=\emptyset$.

2. for $i=1$ to $\text{length}(N)$

if $H[i]=0$ and $N[i]\neq 0$ then

$G=G\cup\{P, P'\}$

*/*such as $N[i]$ is the occurrence probability of the Arabic bi-phoneme PP' in the reference phonetic model and $H[i]$ is the occurrence probability of the same bi-phoneme in the phonetic model proper to a speaker*/*

3. for each phoneme P in G

if $|\text{nbr}(P)/69|=1$ then $M=M\cup\{P\}$

We note by:

- $\text{nbr}(P)$ the repetition number of the phoneme P in the set G .
- 69 the result of $35*2-1$, which is the number of all possible combinations of an Arabic phoneme with all other phonemes of the Arabic alphabet including the phoneme itself.
- M the set that contains mispronounced (substituted) phonemes.

2.2.2 Substituent Sounds

This subsection is dedicated to identify the substituent phonemes. The main idea is that the sum of occurrence probabilities of bi-phonemes containing a wrongly pronounced phoneme is distributed to bi-phonemes containing substituent phonemes. Algorithm 3 is used to assure this task:

Algorithm 3. Detecting substituent sounds

1. We propose these two sets B and R with $B=R=\emptyset$.

2. for $i=1$ to $\text{length}(N)$

if $H[i]+(\text{Avg}-\delta)<H[i]$ then

$B=B\cup\{P, P'\}$

*/*such as $N[i]$ is the occurrence probability of the Arabic bi-phoneme PP' in the reference phonetic model and $H[i]$ is the occurrence probability of the same bi-phoneme in the phonetic model proper to a speaker*/*

3. for each phoneme P in B

if $|\text{nbr}(P)/69|=1$ then $R=R\cup\{P\}$

We note by:

- $\text{nbr}(P)$ the repetition number of the phoneme P in the set B .
- 69 the result of $35*2-1$, which is the number of all possible combinations of an Arabic phoneme with all other phonemes of the Arabic alphabet including the phoneme itself.
- R the set that contains mispronounced (substituted) phonemes.
- $\delta = \text{Standard Deviation}\{N[i], 1 \leq i \leq \text{length}(N), \text{ with } H[i] = 0 \text{ and } N[i] \neq 0\}$.
- $\text{Avg} = \text{The Average}\{N[i], 1 \leq i \leq \text{length}(N), \text{ with } H[i] = 0 \text{ and } N[i] \neq 0\}$.

2.3 Pathological Speech Correction

In this subsection, we give details of the proposed algorithms addressing the correction of pathological speeches. The correction of mispronunciations contained in this class follow the next algorithms:

2.3.1 Tracing Algorithm

The tracing procedure allows drawing the tree representing all possible pronunciations coming from the input:

Algorithm 4. Tracing algorithm (m the pronunciation to be corrected)

```

Begin
if the first phoneme  $P_1 \in R$  then
*the root is labeled NULL
* $P_1$  and its correspondent in  $M$ ,  $P'_1$  form the two sons of this root
*arcs from root to  $P_1$  and  $P'_1$  will be equal-probably weighted
else
* $P_1$  is the root of this tree
End if
for  $2 \leq i \leq |m|$ 
if  $P_{i-1} \in R$  then
* $P_i$  and its correspondent in  $M$ ,  $P'_i$  form the two sons of  $P_{i-1}$ 
*arcs  $P_{i-1}P_i$  and  $P_{i-1}P'_i$  will be weighted respectively by  $P(P_{i-1}P_i)$  and  $P(P_{i-1}P'_i)$ 
if  $P_{i-1} \in R$  ( $P'_{i-1}$  is its correspondent in  $M$ ) then
* $P_i$  and  $P'_i$  form also the two sons of  $P'_{i-1}$ 
*arcs  $P'_{i-1}P_i$  and  $P'_{i-1}P'_i$  will be weighted respectively by  $P(P'_{i-1}P_i)$  and  $P(P'_{i-1}P'_i)$ 
End if
else
* $P_i$  is the single son of  $P_{i-1}$ 
* arc  $P_{i-1}P_i$  will not be weighted (because it is the single path)
if  $P'_{i-1} \in R$  ( $P'_{i-1}$  is its correspondent in  $M$ ) then
* $P_i$  forms also the son of  $P'_{i-1}$ 
* arc  $P'_{i-1}P_i$  will not be weighted (because it is the single path)
End if
End if
End for
End

```

2.3.2 Routing Algorithm

The routing procedure permits browsing the tree generated by the procedure tracing to find the list of Arabic words that correspond to the false pronunciation. A description of this algorithm is as follows:

Algorithm 5. Routing algorithm

Begin

**Depth-first path of tree representing the pronunciation to be corrected (m) by following the arc with the highest weight until arriving at the leaf level.*

**If the obtained path forms one pronunciation m' existing in the lexicon then m' is the correct pronunciation of m*

else

**go back until you get the branch with largest height,*

**delete this branch,*

**Repeat the routing from the actual position.*

End if

End

2.3.3 Practical Examples

First example:

Let us take the example of the correction of the pronunciation “حُطَّلَة”، where:

- $M = \{خ، ص، غ\}$,
- $R = \{ح، ث، ر\}$,

The tracing algorithm will generate the tree as shown in Fig. 3. We apply thereafter the routing algorithm to obtain the correct pronunciation “حُطَّلَة” (the colored branch).

Second example: Let us take the example of the correction of the pronunciation “ذَمِيل”، where:

- $M = \{خ، ص، ج، ز\}$,
- $R = \{ح، ذ، ث\}$,

The tracing algorithm will generate the tree as shown in Fig. 4. We apply afterwards the routing algorithm to obtain the correct pronunciations “زَمِيل” or “جَمِيل” (the colored branches). In such a case, a semantic analysis is required to select the adequate correction.

Third example:

Let us take the example of the correction of the pronunciation “نَحْلَة”، where:

- $M = \{خ، ص، س\}$,
- $R = \{ح، ث\}$,

Applying the routing algorithm, we note that the input pronunciation can be corrected (accepted by the lexicon) and can be also the correct pronunciation of “نَحْلَة”. The colored branches in Fig. 5 present the accepted pronunciations.

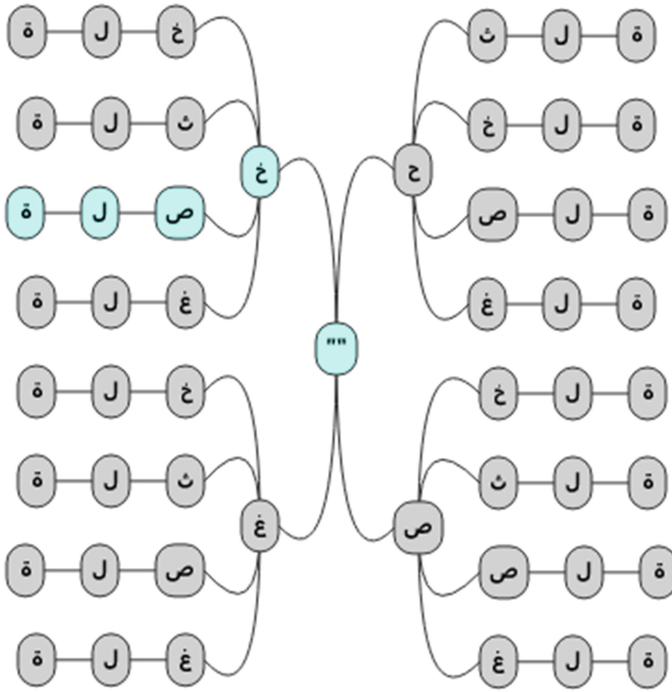


Fig. 3. Correction of the Arabic mispronunciation “حذرة” (Color figure online)

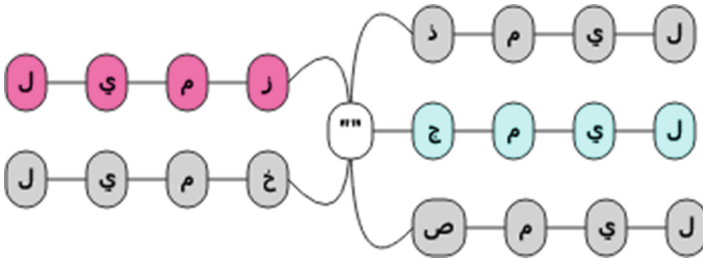


Fig. 4. Correction of the Arabic mispronunciation “ذميلة” (Color figure online)



Fig. 5. Correction of the Arabic mispronunciation “نحلة” (Color figure online)

3 Experimental Results

To realize the proposed method, we use 11 h of healthy speeches to calculate a phonetic model referring to the Arabic language, seven hours of healthy and pathological speeches to train the acoustic model, and 105 speech sequences to test the algorithm of pathological speech correction. The results of speech classification into healthy or pathological are given in Table 1:

Table 1. Obtained results of speech classification

Speech bases	Classification	Speakers	Precision
18 speech sequences	18 pathological	Non-native speakers	95.51% (149 speech sequences correctly classified)
138 speech sequences	51 healthy 87 pathological	Native speakers (healthy and with voice pathologies)	

For pathological speech classification, we launch our proposed algorithm to identify the phonemes that pose mispronunciations contained in the Arabic speech. The identification of problematic phonemes covered different speaker categories. We have obtained an identification rate of problematic phonemes of 96% (average between different categories). In Table 2, we cite different phonemes posing pronunciation defects for each one:

Table 2. Problematic phonemes for different categories of speakers

Speakers	Problematic phonemes	Percent
Native adult speakers	خ غ	36%
	س ص	31%
	ر	17%
	ق ك	11%
	ذ ض ظ	4%
	Others	1%
Native scholar and pre-scholar speakers	ق خ غ	38%
	س ص	30%
	ر	19%
	ض ظ	11%
	Others	2%
	Non-native speakers	ذ ض ظ
ح		28%
قي		23%
خ ع غ		14%
Others		2%

Our system presents a good efficiency of pathological-speech correction. Figure 6 depicts the speech correction rate for each category (native or non-native speakers).

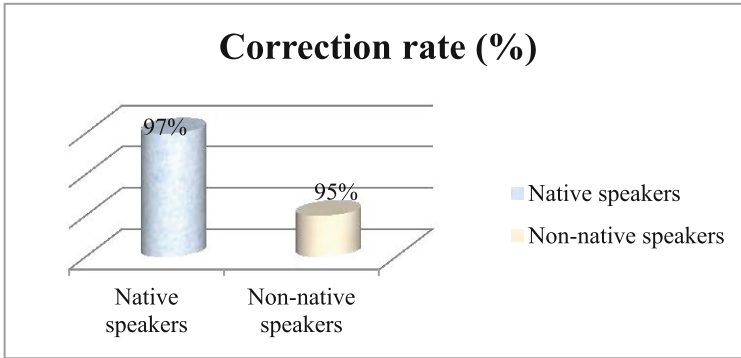


Fig. 6. Pathological speech correction rate

4 Conclusions and Future Work

We can conclude that the comparison between different phonetic models presents a deciding factor to detect vocal pathologies contained in the Arabic speeches and to generate an adequate correction for each input mispronunciation. In this paper, we have proposed a novel algorithm to correct mispronunciations contained in an Arabic acoustic signal. For this purpose, a corpus of 11 h of healthy Arabic speeches has been prepared to calculate the phonetic model referring to the spoken Arabic language. Another corpus containing seven hours of healthy and pathological Arabic speeches has been recorded to train the acoustic model.

Based on 105 speech sequences, our proposed method has attained a high pathological speech correction accuracy. Indeed, we have got 95% as a correction rate for non-native speakers and 97% for native ones.

To the best of our knowledge, this work presents the first attempt addressing the correction of voice pathologies contained in the Arabic speeches. We are satisfied with the obtained results, and our suggested approach can present an important reference for work focalizing on automatic pathological speech processing.

As future work, we can introduce a semantic analyzer to select the appropriate correction of an input mispronunciation for an algorithm of pathological speech correction generating several possible corrections.

References

1. Terbeh, N., Maraoui, M., Zrigui, M.: Probabilistic approach for detection of vocal pathologies in the arabic speech. In: Gelbukh, A. (ed.) *CICLing 2015*. LNCS, vol. 9042, pp. 606–616. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-18117-2_45
2. Alghamdi, M., Almuhtasib, H., Elshafei, M.: Arabic phonological rules. *King Saud Univ. J. Comput. Sci. Inf.* **16**, 1–25 (2004)
3. Patane, G., Russo, M.: The enhanced LBG Algorithm. *Neural Netw.* **14**(9), 1219–1237 (2001)
4. Bréhilil, L., Gascuel, O.: Modèles de Markov caches et apprentissage de sequences
5. Majidnezhad, V., Kheidorov, I.: An ANN-based method for detecting vocal fold pathology. *Int. J. Comput. Appl.* **62**(7), 1–4 (2013)
6. Majidnezhad, V., Kheidorov, I.: A HMM-based method for vocal fold pathology diagnosis. *IJCSI Int. J. Comput. Sci. Issues* **9**(6), 135–138 (2012). No 2
7. Paquet, P.: L'utilisation des réseaux de neurones artificiels en finance. Document de recherche n° 1997-1 (1997)
8. Terbeh, N., Zrigui, M.: Vocal pathologies detection and mispronounced phonemes identification: case of arabic continuous speech. In: *LREC 2016*, 23–28 May 2016, Portorož-Slovenia (2016)
9. Hawashin, B., Mansour, A., Aljawarneh, S., Fahmy, A.A., Al Raddady, F., Shrivastava, A., Rajawat, A.S., Malika, C.R., Mishra, S., Yadav, R.N.: An efficient feature selection method for arabic text classification. *Int. J. Comput. Appl.* **83**(17), 1–6 (2013)
10. Zhou, Z., Meng, H.M., Lo, W.K.: A multi-pass error detection and correction framework for Mandarin LVCSR. In: *Proceedings of the International Conference on Spoken Language Processing*, Pittsburgh, PA, pp. 1646–1649 (2006)
11. Kaki, S., Sumita, E., Iida, H.: A method for correcting errors in speech recognition using the statistical features of character co-occurrence. In: *COLING-ACL*, Montreal, Quebec, Canada, pp. 653–657 (1998)
12. Ajibola, A.S., Rashid, N.K.B.A.M., Sediono, W., Hashim, N.N.W.N.: A novel approach to stuttered speech correction. *Jurnal Ilmu Komputer dan Informasi* **9**(2), 80–87 (2016)
13. Bassil, Y., Alwani, M.: Post-editing error correction algorithm for speech recognition using bing spelling suggestion. *Int. J. Adv. Comput. Sci. Appl.* **3**(2), 95–101 (2012)
14. Muhammad, G., Alsulaiman, M., Ali, Z., Malki, K., Mesallam, T., Farahat, M.: Voice pathology detection and classification using auto-correlation and entropy features in different frequency regions. *IEEE Access* **PP**(99), 1 (2017)

Speech Analysis



Identification of Pronunciation Defects in Spoken Arabic Language

Naim Terbeh^(✉) and Mounir Zrigui

LaTICE Laboratory, Faculty of Sciences of Monastir, 5000 Monastir, Tunisia
naim.terbeh@gmail.com, terbehnaim1987@gmail.com,
mounir.zrigui@fsm.rnu.tn

Abstract. The detection of vocal pathologies is one of the novelties addressing automatic speech processing. There are several intervening approaches that are based on features contained in an acoustic signal and on natural language processing techniques. However, up to our knowledge, these studies are not extended to detect phonemes that pose degraded speeches. In this paper, we propose a new method to detect mispronounced sounds. We are based on a phonetic-probabilistic modeling. The invented study accounts four fundamental tasks. The first task summarizes the calculation of the probabilistic-phonetic model referring to Arabic speech. The second one is dedicated to calculate the probabilistic-phonetic model appropriate to a speaker whose elocution is classified as pathological. Thirdly, we compare between the two previous models to distinguish two main classes: the input speech can be healthy or pathological. The fourth stage consists in introducing an original algorithm based on a phonetic modeling to generate problematic sounds and to evaluate the elocution of each speaker having voice pathologies by attributing them a language level. This task will be only applied if the input speech is pathological. The obtained results are satisfactory. We have attained a problematic-sound identification rate of 96%.

Keywords: Problematic sounds · Arabic healthy/pathological speech
Probabilistic-phonetic modeling · Pronunciation defects · Elocution evaluation

1 Introduction and State of the Art

The voice commands are in progressively used in human-machine communication, which imposes new applications [8, 10]. Introducing probabilistic and linguistic approaches can enormously facilitate human-machine dialogues. Nevertheless, the important number of people suffering from language disabilities presents one of the main obstacles for the propagation of this communication mode. In this context, speech therapy, biology and computer sciences are needed for the introduction of new approaches to the detection of vocal defects contained in speech signals. Yet, most work are not extended to identify problematic sounds that pose pronunciation defects and to evaluate produced pronunciations classified as pathological, which is the objective of the present study.

An acoustic signal can present different types of pronunciation defects. We distinguish five principal categories [13]:

- Suppression: This handicap will appear if the speaker removes one or more sounds from the desired pronunciation. For example, speakers can produce the mispronunciation “صوّة” instead of the wanted “صورة” by removing the sound “ر”.
- Substitution: This voice defect manifests when the speaker replaces one or more desired sounds by others. For instance, he produces the bad pronunciation “مقنّ” instead of “مقّصّ”, replacing the morpheme “ص” by “نّ”.
- Distortion: It is a manifestation of substitution. It will display when the desired morpheme is replaced by another similar sound like permutation between “نّ” and “ط”, between “ض” and “ظ”, between “س” and “ص”, etc. As an example, the speaker may produce the mispronunciation “تاقّة” instead of the desired “طاقّة”, replacing the sound “ط” by “ت”.
- Permutation: This vocal pathology is famous in children speech. It will occur if speakers switch between two phonemes of the desired pronunciation; e.g., speakers can produce the mispronunciation “السمش” instead of the desired “الشمس”, permuting the two phonemes “ش” and “س”.
- Addition: This voice handicap can be expressed by the addition of one or more sounds to the wanted pronunciation. The famous example is that when speakers add the sound “ن” to all pronunciations that finish by the “Mad/مد” (“ا” or “ى”, “ي” and “و”).

A single pronunciation can presents two or more different categories of voice disorders. For instance, the mispronunciation “الشمش” illustrates a permutation between the two sounds “نّ” and “ش” and a replacement of the phoneme “س” by “نّ”.

There is some work which addresses the identification of voice defects and the pronunciation evaluation. We can cite:

- Voice pathology detection
 - Alsulaiman et al. suggested a new solution to extract accurate and robust features for detecting and classifying voice anomalies contained in the Arabic speech. This proposition attained a high performance based on the autocorrelation and the entropy in different frequency bands. It was the objective of the work in [17].
 - Terbeh et al. invented in [1] a novel method whose aim was to detect voice disorders that would figure in Arabic speech. In this study, the authors based their study on probabilistic-phonetic modeling.
 - In [7], Majidnezhad et al. recommended a new algorithm to detect voice handicaps and to classify acoustic signals into healthy or pathological. For this objective, the authors based their work on artificial neural networks [11].
 - Majidnezhad et al. created in [9] a new solution to detect degraded frames from speech signals. For this objective, the authors needed to utilize the hidden Markov models [6] and the Linde-Buzo-Gray algorithm [5].

- Pronunciation evaluation
 - Lin et al. invented in [18] a multi-level pronunciation evaluator for non-native Mandarin tones. The authors proposed two different models: a referenced tone model and a model appropriate to one speaker. Comparing between these models generated different confidence measures of pronunciation evaluation: log posterior ratios, average frame-level log posteriors and segment-level log posteriors.
 - The work in [19] presented a new proposition invented by Patil et al. in order to extract acoustic-phonetic characteristics that would be used to separate non-native speakers from native ones. This contribution showed a good performance in the evaluation of spoken Hindi language.
 - Belgacem in [14] suggested in his PhD thesis an original approach to assist francophone learners of the spoken Arabic language in improving their pronunciations. The author proposed as a preliminary task the evaluation of the pronunciation for each learner of the spoken Arabic language.

The previously cited studies did not solve the problems concerning the localization of voice disorders by identifying the problematic sounds. Also, they were not extended to generate an evaluation of pronunciations by attributing an elocution level for each speaker.

Consequently, this work invents an original study based on probabilistic-phonetic modeling that addresses the identification and localization of problematic sounds which generate a degraded Arabic speech. For each speaker with a voice handicap, we use the forced alignment score in order to calculate their pronunciation level.

The structuration of this paper is as follows. First, a description of our contribution takes place in Sect. 2. Then, we draw the experiment results in Sect. 3. Finally, the conclusions and future work are mentioned in Sect. 4.

2 Contribution

The principal intervention is to propose a new algorithm based on probabilistic-phonetic modeling to localize problematic sounds that pose voice disorders. For each speaker with voice defects, an original algorithm will be applied to identify and to localize the problematic phonemes that beget pronunciation defects.

Our contributions are explained in the following points:

- Calculating a probabilistic-phonetic model referring to Arabic speeches
- Calculating a probabilistic-phonetic model appropriate for each speaker
- Distinguishing between healthy and pathological speeches
- Identifying phonemes begetting a degraded speech
- Evaluating pronunciation

In Fig. 1, we describe the operation of the proposed system:

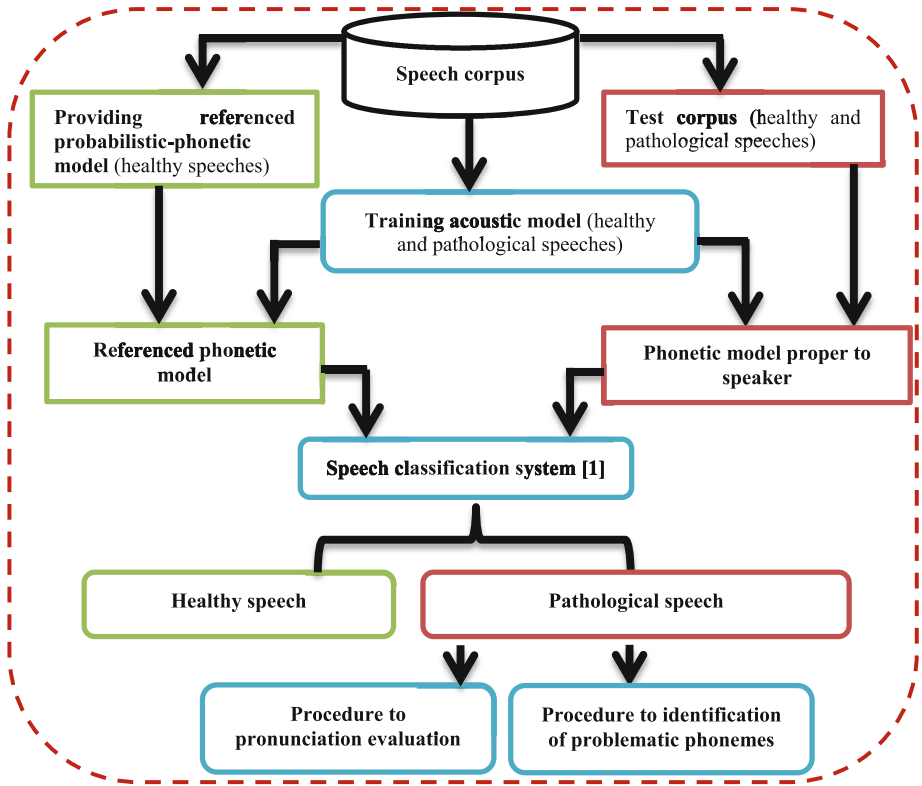


Fig. 1. General form of proposed system

The previously cited requirements will be explained in the next paragraphs.

2.1 Scalar Product in \mathfrak{R}^n

The scalar product in \mathfrak{R}^n noted by $\langle x|y \rangle$ is the function which associates to the vectors $x = (x_1, x_2, \dots, x_n) \in \mathfrak{R}^n$ and $y = (y_1, y_2, \dots, y_n) \in \mathfrak{R}^n$ the quantity:

$$\langle x|y \rangle = \sum_{i=1}^n x_i y_i \tag{1}$$

This quantity can also be expressed as follows:

$$\langle x|y \rangle = \|x\| \cdot \|y\| \cdot \cos(\theta) \tag{2}$$

We note by:

- θ the angle which separates the two vectors x and y .
- $\|x\|$ the norm of the vector x , $\|x\| = \sqrt{x_1^2 + x_2^2 + \dots + x_n^2}$.
- $\|y\|$ the norm of the vector y , $\|y\| = \sqrt{y_1^2 + y_2^2 + \dots + y_n^2}$.

Thereby, we can deduce that:

$$\cos(\theta) = \frac{\sum_{i=1}^n x_i y_i}{\|x\| \cdot \|y\|} \tag{3}$$

If we can generate a vectorial representation in \mathfrak{R}^n for each produced speech signal, the angle θ can be used as a measure of similarity between the speech signals represented by these vectors. Indeed, the similarity between different acoustic signals is proportional to the angular distance (the angle θ) which separates between the representative vectors.

2.2 Probabilistic-Phonetic Modeling and Speech Classification

In this subsection, we describe the procedures used to calculate the probabilistic-phonetic model and to classify acoustic signals as healthy or pathological. The phonetic model is defined by the vector whose coefficients present the probability occurrences of each Arabic bi-phoneme. In Fig. 2, we present the procedure followed to calculate the probabilistic-phonetic model and an extract from the model referring to the Arabic speech.

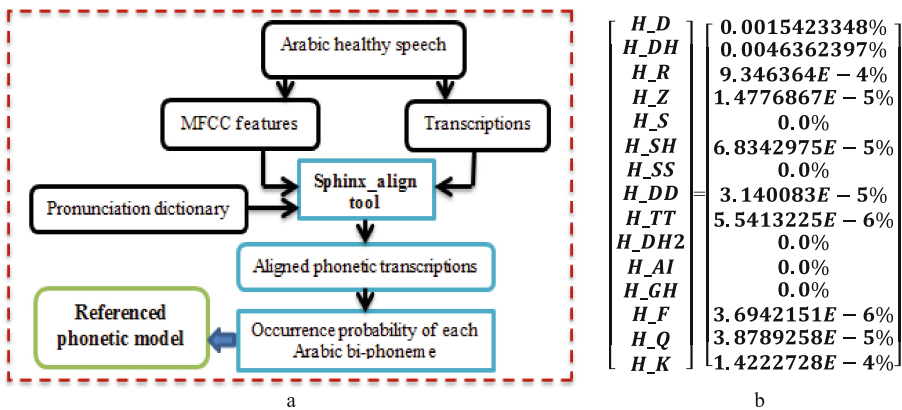


Fig. 2. (a) Process providing the probabilistic-phonetic model; and (b) extract from obtained model

The distinction between healthy and pathological speeches follows three fundamental steps. First, we calculate a probabilistic-phonetic model (vector in \mathfrak{R}^{1225}) referring to the Arabic speech, so we need to train an Arabic acoustic model and to prepare a large healthy speech corpus (checked in by native ant healthy speakers).

Second, we utilize for each speaker the trained acoustic model to generate a probabilistic-phonetic model summarizing the produced speech. Finally, the angle that separates these two models results in a decision about the produced speech (healthy or pathological) as defined in [1].

A comparison between the latter calculated models assures also the identification of the problematic phonemes begetting degraded speeches.

2.3 Identification of Problematic Phonemes

During the current section, we note by:

- N the vector summarizing the probabilistic-phonetic model referring to the standard spoken Arabic language.
- H the vector expressing the probabilistic-phonetic model appropriate to a speaker.

In this subsection, and for each pathological speech frame, we identify the phonemes that beget pronunciation disorders. We consider that a mispronounced sound does never appear in the phonetic transcription summarizing the produced speech. Therefore, we have a coefficient equal to zero in the phonetic model for all bi-phonemes containing a problematic phoneme.

The first algorithm is based on the comparison between N and H vectors to pinpoint problematic phonemes.

Algorithm 1. Identification of problematic phonemes

1. We propose these two sets M and G with $M=G=\emptyset$.

2. for $i=1$ to $\text{length}(N)$

if $H[i]=0$ and $N[i]\neq 0$ then

$G=G\cup\{P, P'\}$

*/*such as $N[i]$ is the occurrence probability of the Arabic bi-phoneme PP' in the phonetic model referring Arabic speeches and $H[i]$ is the occurrence probability of the same bi-phoneme (PP') in the phonetic model appropriate to a speaker*/*

3. for each phoneme P in G

if $\lfloor \text{nbr}(P)/69 \rfloor = 1$ then

$M=M\cup\{P\}$

We note by:

- $\text{nbr}(P)$ the repetition number of the phoneme P in the set G .
- 69 the result of $35 * 2 - 1$, which is the number of all possible combinations of an Arabic phoneme with all other phonemes of the Arabic alphabet.
- M the set containing mispronounced sounds.

2.4 Forced Alignment Score and Elocution Evaluation

The forced alignment consists in aligning an acoustic signal with its corresponding transcription and to affecting to each phoneme a score (forced alignment score) according to the distance separating it from the norm (acoustic model). As a support to

the sphinx-align tool, we realize this alignment of the speech signal with an adequate phonetic transcription.

In order to evaluate the quality of pronunciation of speakers, we propose an algorithm based on the forced alignment score that compares between the pronunciation produced by a speaker having elocution problems and a reference pronunciation (checked in by a native and healthy speaker).

The proposition consists in calculating the average (Avg) of the forced alignment scores calculated for different native speakers and the standard deviation “ δ ” which separates between these scores. Then we can distinguish three levels of pronunciation, as explained in Algorithm 2:

Algorithm 2. Pronunciation evaluation

For every speaker with voice defects

For each problematic phoneme P_i

1. We calculate the forced alignment score $F(P_i)$

2. We distinguish three cases:

* if $(F(P_i) > \text{Avg} + \delta)$, then the pronunciation is *Good*

* if $(\text{Avg} - 2 * \delta < F(P_i) < \text{Avg} + \delta)$, then the pronunciation is *Medium*

* if $(F(P_i) < \text{Avg} - 2 * \delta)$, then the pronunciation is *Bad*

3 Experiments

To realize the proposed system of problematic-phoneme identification and pronunciation evaluation:

- 35 Arabic phonemes are used [2].
- 11 h of healthy Arabic speech are utilized in calculating the probabilistic-phonetic model referring to the spoken Arabic language.
- Seven hours of Arabic speech (healthy and pathological speeches, native and non-native speakers) are used in training our acoustic model.
- A speech base is in *.wav format and in mono speaker mode.
- The test base is created with the assistance of a speech therapist. It contains 105 pathological speech frames.
 - 87 audio sequences are checked in by native speakers suffering from voice pathologies.
 - 18 audio sequences are checked in by non-native speakers (learners of Arabic vocabulary).

3.1 Obtained Results

- **Speech classification**

For this task we use the system elaborated by Terbeh et al. (described in [1]) to classify 156 speech sequences. Table 1 provides the attained results.

Only pathological speech sequences will be used to identify problematic sounds.

- **Problematic phonemes identification**

For pathological speeches, we launch a novel algorithm (defined in Algorithm 1) to identify the sounds begetting degradations in the Arabic speech. The identification of problematic phonemes cover different speaker categories. We get an identification rate of 96%. In Tables 1, 2, 3 and 4, we cite the different phonemes posing pronunciation disorders per category:

Table 1. Obtained results of speech classification

Speech bases	Classification	Speakers	Precision
18 speech sequences	18 pathological	Non-native speakers	95.51% (149 speech sequences correctly classified)
138 speech sequences	51 healthy 87 pathological	Native speakers (healthy and with voice pathologies)	

Table 2. Problematic phonemes for native speakers (adults)

Arabic problematic phonemes	Rate of pronunciation disorders
خ غ	36%
ص س	31%
ر	17%
ق ك	11%
ذ ض ظ	4%
Others	1%

Table 3. Problematic phonemes for native speakers (scholar and pre-scholar speakers)

Arabic problematic phonemes	Rate of pronunciation disorders
ق خ غ	38%
ص س	30%
ر	19%
ض ظ	11%
Others	2%

Table 4. Problematic phonemes for non-native speakers

Arabic problematic phonemes	Rate of pronunciation disorders
ذ ض ظ	33%
ح	28%
ق	23%
خ ع غ	14%
Others	2%

• Pronunciation evaluation

In order to follow the progression of the speaker elocution (specifically of learners of Arabic vocabulary), we launch for each pathological sequence an algorithm (defined in Algorithm 2) permitting the evaluation of produced pronunciation. Table 5 describes the elocution progression for native and non-native speakers compared to the referenced (native and healthy speaker) in seven successive tests (one test after one week of training).

Table 5. Evaluation of pronunciations

	Elocution evaluation						
	1 st test	2 nd test	3 rd test	4 th test	5 th test	6 th test	7 th test
Native speakers	Bad	Bad	Bad	Bad	Bad	Bad	Medium
Non-native speakers	Bad	Bad	Bad	Medium	Medium	Medium	Medium

3.2 Discussion

For native adult speakers, the phonemes generating a degraded speech are caused either by language disabilities or by a difficulty in mastering some phoneme pronunciations for specific situations in Arabic speeches.

For foreign speakers, the phonemes that pose pronunciation defects are often similar to other phonemes in their native languages; speakers are often hampered by phonemes in maternal languages.

The obtained results for the scholar and pre-scholar categories of speakers confirm the hypothesis which says that in this age speakers are not able to master phonemes engendering a vibration of vocal cords.

Non-native speakers (learners of Arabic vocabulary) present a good progression in learning problematic phonemes compared to native speakers (speakers with voice pathologies).

4 Conclusions and Future Work

To conclude, we can mention that the natural language processing techniques present an adequate method in spoken-language diagnostics to identify the problematic phonemes that pose pronunciation disorders for speakers of the Arabic language.

The experiment results have shown that the proposed approach has a high phoneme-identification accuracy and a precise follow-up of pronunciation progress. Indeed, we have attained 96% as an identification rate of problematic phonemes and the progression of elocution is validated by a speech therapist. Thanks to these encouraging results, our suggested method can be an important reference for future work focalizing on pathological speech analysis.

Supporting this work, we can elaborate new algorithms addressing pathological speech correction [3, 4] and non-native speaker assistance to learn the Arabic vocabulary [12]. It is possible also to extend this suggested method to other studies addressing speakers and dialect identification [15].

References

1. Terbeh, N., Maraoui, M., Zrigui, M.: Probabilistic approach for detection of vocal pathologies in the arabic speech. In: *CICLing 2015, Cairo-Egypt*, 14–20 April 2015 (2015)
2. Alghamdi, M., Almuhtasib, H., Elshafei, M.: Arabic phonological rules. *King Saud Univ. J. Comput. Sci. Inf.* **16**, 1–25 (2004)
3. Terbeh, N., Labidi, M., Zrigui, M.: Automatic speech correction: a step to speech recognition for people with disabilities. In: *ICTA 2013, Hammamet-Tunisia*, 23–26 October 2013 (2013)
4. Terbeh, N., Zrigui, M.: *Vers la Correction Automatique de la Parole Arabe*. Citala 2014, Oujda-Morocco, 26–27 November 2014 (2014)
5. Patane, G., Russo, M.: The enhanced LBG algorithm. *Neural Netw.* **14**(9), 1219–1237 (2001)
6. Bréhilil, L., Gascuel, O.: *Modèles de Markov caches et apprentissage de séquences*
7. Majidnezhad, V., Kheidorov, I.: An ANN-based method for detecting vocal fold pathology. *Int. J. Comput. Appl.* **62**(7), 1–4 (2013)
8. Karsenty, A.: *Le collecticiel: De l'interaction homme-machine à la communication homme-machine-homme* (1994)
9. Majidnezhad, V., Kheidorov, I.: A HMM-based method for vocal fold pathology diagnosis. *IJCSI Int. J. Comput. Sci. Issues* **9**(6), 2 (2012)
10. Daniel, S.: *De l'interaction homme-machine individuelle aux systèmes multi-utilisateurs (L'exemple de la communication homme-homme médiatisée)*. Doctoral thesis (1995)
11. Paquet, P.: *L'utilisation des réseaux de neurones artificiels en finance*. Document de recherche n° 1997-1 (1997)
12. Terbeh, N., Zrigui, M.: Vocal pathologies detection and mispronounced phonemes identification: case of Arabic continuous speech. In: *LREC 2016, Portorož-Slovenia*, 23–28 May 2016 (2016)
13. <http://kenanaonline.com/users/dkkhaledeInagar/photos/1238136361>. Accessed 24 June 2016
14. Belgacem, M.: *Reconnaissance automatique de la parole et ALAO: Vers un système d'apprentissage de l'arabe oral*. Ph.D. thesis (2011)

15. Biadisy, F., Hirschberg, J., Habash, N.: Spoken Arabic dialect identification using phonotactic modeling. In: Proceedings of the EACL 2009 Workshop on Computational Approaches to Semitic Languages, pp. 53–61. Association for Computational Linguistics (2009)
16. Elshafei, M., Almuhtasib, H., Alghamdi, M.: Statistical methods for automatic diacritization of Arabic text. In: Proceedings of 18th National computer Conference, Riyadh, Saudi Arabia (2006)
17. Muhammad, G., Alsulaiman, M., Ali, Z., Malki, K., Mesallam, T., Farahat, M.: Voice pathology detection and classification using auto-correlation and entropy features in different frequency regions. *IEEE Access* **PP**(99), 1 (2017)
18. Lin, J., Xie, Y., Zhang, J.: Automatic pronunciation evaluation of non-native Mandarin tone by using multi-level confidence measures. In: Interspeech 2016
19. Patil, V.V., Rao, P.: Detection of phonemic aspiration for spoken Hindi pronunciation evaluation. *J. Phonetics* **54**, 202–221 (2016)

Author Index

- Agrawal, Samarth 275
Aoki, Tatsuya 3
Apoorva, G. Drushti 41
Asahara, Masayuki 155
Atkinson, Katie 51
- Bhattacharyya, Pushpak 275
Bollegala, Danushka 51, 76
- Carman, Mark J. 275
Chea, Vichet 179
Cho, Seung Woo 127
Coenen, Frans 51
- Ding, Chenchen 179, 191
- Echizen, Isao 288
- Hakami, Huda 76
Han, Buxin 88
Hlaing, Aye Mya 263
- Joshi, Aditya 275
- Kato, Sachi 155
Komiya, Kanako 205
Kotani, Katsunori 329
- Lai, Dac-Viet 249
Laib, Meriama 101
Le Minh, Nguyen 249
Le, Anh-Cuong 28
Le, Thi-Kim-Chung 28
Lee, Eui-Hyeon 127
Lee, Jong-Hyeok 127
Le-Hong, Phuong 219
Li, Liang 115
Li, Minglei 88
Li, Pengyu 115
Liu, Pingping 88
Liu, Yifan 115
Lu, Qin 88
- Mamidi, Radhika 41
Mandya, Angrosh 51, 76
- Miyauchi, Takuya 155
Mon, Aye Nyein 314
- Nakagawa, Natsuko 155
Nasukawa, Tetsuya 3
Nguyen, Huy 15
Nguyen, Le-Minh 63, 137, 233
Nguyen, Minh-Le 15
Nguyen, Truong-Son 233
Nguyen-Son, Hoang-Quoc 288
- Okumura, Manabu 3
- Pa, Win Pa 191, 263, 303, 314
Pham, Duc-Hong 28
Pham, Thai-Hoang 219
Pluempitwiriwawej, Charnyote 169
- Qin, Zengchang 115
- Sam, Sethserey 179
Sangvat, Sokunsatya 169
Sasaki, Minoru 205
Semmar, Nasredine 101
Seng, Sopheap 179
Shinnou, Hiroyuki 205
Soe Naing, Hay Mar 303
Son, Nguyen Truong 249
Sumita, Eiichiro 179, 191
- Takamura, Hiroya 3
Terbeh, Naim 341, 355
Thu, Ye Kyaw 263, 314
Tran, Van-Khanh 63
Trieu, Hai-Long 137
- Utiyama, Masao 179, 191
- Wan, Tao 115
- Yoshikawa, Katsumasa 3
Yoshimi, Takehiko 329
- Zrigui, Mounir 341, 355