Naive Bayes and BiLSTM Ensemble for Discriminating between Mainland and Taiwan Variation of Mandarin Chinese

Li Yang
Tongji University
Shanghai
China
li.yang@tongji.edu.cn

Yang Xiang
Tongji University
Shanghai
China
shxiangyang@tongji.edu.cn

Abstract

Automatic dialect identification is a more challenging task than language identification, requiring ability to discriminate between different varieties within the same language family. In this paper, we propose an ensemble based system, which combine traditional machine learning models trained on bag of n-gram fetures with deep learning models trained on word embeddings, to solve the Discriminating between Mainland and Taiwan Variation of Mandarin Chinese (DMT) shared task at VarDial 2019 . Our experiments show that a character bigram-trigram based Navie Bayes is a very strong model for identification varieties of Mandarin Chinense. By combine Navie Bayes and BiLSTM using average ensembling, a simple yet effective approach, our system achived a accuracy acore of 92%, ranking second out of 12 teams.

1 Introduction

Dialect identification can be considered as a specical case of language identification which aims at distinguishing related languages or variations of a specific language. Being capable of detecting dialect accurately is an important step for many natural language processing piplines and applications, such as automatic speech recognition, machine traslation, multilingual data acquisition, health monitoring and so on. While language identification can already achieve relatively high performance with a simple model, dialect identification is still a tough probelm remained to be tackled. In contrast to the language identification scenario, the linguistic difference between related languages is less obvious, because dialects typically share a common phonetic inventory and word vocabularies. For that reason, dialect identificaion has drawn many researchers' attention in recent years, becoming an important and attractive research topic.

Term	Mainland China	Taiwan
tomato	番茄	西柿
taxi	出租车	程
bicycle	自行车	踏
program	程序	程式
kindergarten	幼儿园	幼稚

Table 1: Different expressions with the same meaning used in Mainland China and Taiwan.

Mandarin Chinese is a group of realted varieties of Chinese spoken across many different regions. The group includes *Putonghua*, the offical language of Mainland China, and *Guoyu*, another Mandarin variant widely spoken in Taiwan. However related they are, there are still some difference between these two varieties. First, the most notable one is the character set they use. Taiwanese Mandarin uses traditional Chinese characters, as opposed to the simplified Chinense characters on the mainland. Take the phrase "*natural language processsing*" as an example, its simplified character form adopted in Mainland China is "自然语言处理", while Taiwan uses the traditional character form "自然語言處理". Second, . some vocabularies differ. There are some terms can be understood by both varieties to mean the same thing. However, their preferred usage distinguishs. Table 1 lists some examples. Apart from character form and vocabularies, the pronunciation escpecially in terms of tone is also different. But we don't discusss in this paper since it's less relevant.

The DMT task, first introduced by VarDial evalution campagin this year, aims at determining whether the sentence belongs to news articles from Mainland China or from Taiwan. The organizers prepared two tracks, for both traditional and simplified by converting one to the other, and asked participants to predict the labels for unified text instances in both tracks. For that reason, we can't make use of character form to discriminate these two language varieties. Mainstream approaches to dialect identification is to regard it as a text classification task and use linear support vector machines (SVMs) with bag of word and character n-grams to solve the task. However, we want to know what's the best classification algorithm for DMT task. Therefore, we experimented with serveral classical machine learning models trained on different word or character level n-gram fetures and fetures combinaton. Besides, deep learning approaches have achieve a lot of sucess in many kinds of nlp tasks including question answering (Feng et al., 2015), sentiment analysis (Chen et al., 2017a), machine translation (Vaswani et al., 2017a), natural language inference (Chen et al., 2017b). To evaluate how much can deep neural neworks can help identify language varieties, we test 7 different deep learning models, varying from CNN based models, RNN based models to CNN-RNN hybrid models. Fully performance comparison to machine learning models is conducted. Finally, we explore different ways to combine the classifiers we discuss before.

The rest of this paper is organized as follows. Section 2 contains related work. In Section 3, we describe the dataset and our proposed solution. Experimental results and discussiona are presented in Section 4, and we give our conclusion in Section 5.

2 Related work

A number of works have devoted to identify different language varieties or related language, especially since the series of VarDial evalution campaigns (Malmasi et al., 2016; Zampieri et al., 2017, 2018). (Lui and Cook, 2013) study on English dialect identification and present serveral classification approaches to classify Australia, British and Caniadian English. (Zampieri and Gebre, 2012) utilize a character n-gram and a word n-gram language model for the automatic classificaton of two written varieties of Portuguese: European and Brazilian. (Ciobanu and Dinu, 2016) conduct an intial study on the dialects of Romanian and proposed using the orthographic and phonetic features of the words to build a dialect classifier. (Clematide and Makarov, 2017) use a majority-vote ensemble of the Navie Bayes, CRF and SVM systems for Swiss German dialects identification. (Kreutz and Daelemans, 2018) uses two SVM classifiers:one trained on word n-grams fewtures and one trained on Pos n-grams to determine a document to be in Flemish Dutch or Netherlandic Dutch. (Çöltekin et al., 2018) use a unified SVM model based on character and word n-grams features with careful hyperparameter tuning for 4 language/dialect identification task.

However, methods to discriminate between varieties of Mandarin Chinese haven't been well studied. (Huang and Lee, 2008) use a top-bag-od-word similarity based contrastive approach to reflect distance among three varieties of Mandarin: Mainland CHina, Singapore and Taiwan. (Xu et al., 2016) deal with 6 varieties of Mandarin: Maninland, Hong Kong, Taiwan, Macao, Malaysia and Singapore. They discover that character bigram and word segmentation based feature work better than traditional character unigram, and some features such as character form, PMI-based and word alignment-based features can help improve performance. A thorough comparison of different architectures and algorithms has yet to be conducted.

3 Data and Methodology

3.1 Data

The DMT task is provided with labeled sentences belongs to news articles from Mainland China or from Taiwan, which compose of 18770 instances for training set, 2000 for validation set and 2000 for test set. As shown in table 2, the DMT data set has a perfectly balanced label distribution. The avergae sentence length (in word level) of two variety are almost the same, both of them are short text. It's worth mentioning that the organizers have prepared two data set: traditional and simplified version by converting simplified to traditional, traditional to simplified charcters, which means we can't utilize character form feature to help us discriminate between theses two language variety. We don't apply any preprocessing since the sentences have been tokenized and punctuation removed from the texts.

3.2 Machine Learning Models with n-gram features

Machine learning models based on feature engineering are the most common methods for dialect identification. In this paper, we experiment with

Variety	Number of instances			Sentence length			
	train	valid	test	min	max	avg	st.dev.
Mainland China	9385	1000	1000	5	66	9.63	3.73
Taiwan	9385	1000	1000	6	48	9.24	3.30

Table 2: Staticstics of data set for each variety. Sentence lengths are calculated based on word-level tokens from training and validation set.

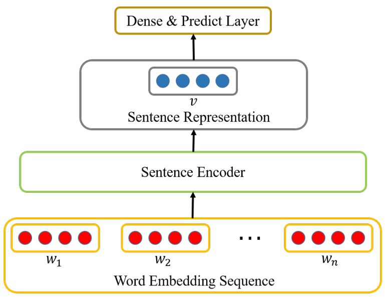

Figure 1: The overall framework of ALA model.

3 different classifiers: (1) logistic regression (LR), (2) linear support vector machine (SVM), and (3) multinomial Naive Bayes (MNB) based on bag of n-gram features. We also examine other Navie Bayes models such as Gaussian Navie Bayes and Bernoulli Naive Bayes, but they are inferior to multinomial Naive Bayes on the validation set for DMT task. The bag of n-gram features include word and character level n-gram with size ranging from unigram to n-gram of one specific order. We conduct a number of experiments to fully explore the most contriuting features and feature cobinations for DMT task, and the results are shown in next section.

3.3 Deep Learning Models with word embeddings

Deep neural networks (DNNs) are of growing interest for their capacity to learn text sentence representation from data without careful engineering of features. For short-text classification task, Convolution neural network (CNN) and recurrent neural network (RNN) are two mainstream DNN architectures. In this paper, we propose a number of deep models based on the same framework to solve DMT task. Figure1 shows a high-level view of the architecture. Vectically, thr figure depicts 3 major components: (1) **Input Embedding Layer**. Suppose the sentence has n tokens, we use a pre-trained embedding method Word2vec (Mikolov et al., 2013) based on training data to represent it in a sequence of word embeddings: $S = (\mathbf{w_1}, \mathbf{w_2}, \cdots \mathbf{w_n})$. Here w_i is a vector standing for a d dimentional word embedding for th i-th word in the sentence. S is thus a sequence that concatenates all word embeddings together. We did try to use character embeddings as input and other pre-trained embedding method such as Glove (Pennington et al., 2014) and Fasttext (Bojanowski et al., 2017) but observed no further improvement on validation set. (2) **Sentence Encoder Layer**. Sentence encoder specified by different models processes the input word embeddings sequence and output a sentence representation: $v = encode\left(S\right)$. (3) **Output Layer**. After obtaining sentence vector, we feed it through one hidden dense layer with 256 unit and a final predict dense layer: $\hat{y} = \sigma\left(\mathbf{W_p}\gamma\left(\mathbf{W_h}v + \mathbf{b_h}\right) + \mathbf{b_p}\right)$, where $\mathbf{W_h}$ and $\mathbf{b_h}$ are the parameters for hidden layer, $\mathbf{W_p}$ and $\mathbf{b_p}$ are the parameters for predict layer, γ and σ are relu and sigmoid activation function respectively, $\hat{\mathbf{y}} \in \mathbb{R}$ represent the predicted score for postive class. During traing process, we minimize the binary cross-entropy loss defined as follow:

$$L = -\frac{1}{N}\sum_{i=1}^{N}\left(y_i \log \hat{y}_i + (1 - y_i)\log\left(1 - \hat{y}_i\right)\right) \tag{1}$$

where y_i is the ground-truth.

We compare 7 different deep architectures to encode sentences into fixed-size representation, varying from cnn-based neural networks, rnn-based neural networks to cnn-rnn hybird neural network.

- **CNN:** First introduced by (Kim, 2014), the convolution network applies a concolution peration with a filter $\mathbf{w} \in \mathbb{R}^{hd}$ to a window of h words to produce a new feature.

$$c_i = f\left(\mathbf{w} \cdot \mathbf{x}_{i:i+h-1} + b\right) \tag{2}$$

By applying this filter to each possible window of words in the sentence, a feture map can be produced. In this paper, we use 300 filters with size ranging from 2 to 5 to extract four 300 dimensional feature maps. After that, we apply max-over-time pooling operation by taking the highest value for each feature map to capture the most important feature, then concatanate all the fewture to represent the input sentence.

- **DCNN:** (Kalchbrenner et al., 2014) use a dynamic convolution neural network (DCNN) that alternates wide concolution layers and dynamic k-Max pooling layers for sentence modeling. By applying k-Max pooling operation, a feature graph over the sentence can be induced, which is capable of explicitly capturing both short and long-range relations.

- **DPCNN:** (Johnson and Zhang, 2017) propose a deep concolutional neural network by stacking concolution blocks (two concolution layers and a shortcut) interleaved with pooling layers with stride 2 for downsampling. The 2-stride downsampling reduces the size of the internal represenation of each document by half, enables efficient representation of long-range association in the text. And the shortcut connection ensure training of deep networks. DPCNN has been shown powerful in many text classificaiton task.

- **BiLSTM:** LSTM is an effective neural network for sentence modeling for its ability to capture long-term dependencies. BiLSTM use a forward and a backward LSTM to process sequence, such that each produced hidden state can contain information from context in two opposite direction. Specifically, at each time step t, hidden state h_t is the concatenation of results from forward and backward LSTM:

$$\begin{aligned} \overrightarrow{h_t} &= \overrightarrow{\text{LSTM}}\left(w_1, w_2, \ldots, w_t\right) \\ \overleftarrow{h_t} &= \overleftarrow{\text{LSTM}}\left(w_n, w_{n-1}, \ldots, w_t\right) \\ h_t &= \left[\overrightarrow{h_t}, \overleftarrow{h_t}\right] \end{aligned} \tag{3}$$

  After obtaining hidden states squence, we apply max-over-time pooling operation to form a fixed-size vector as sentence representation v.

- **Self-attentive BiLSTM:** Attention mechanism is most comonly used in sequence-to-sequence models to attend to encoder states (Bahdanau et al., 2014; Vaswani et al., 2017b). In this paper, we make use of attention, more specifically, self-attention (Lin et al., 2017) to obtain a distribution over features learned from BiLSTM (a.k.a hidden states). Suppose H is the output hidden vectors of BiLSTM: $H = (\mathbf{h_1}, \mathbf{h_2}, \cdots \mathbf{h_n})$, we can calculate the attention vector α and the final sentence representation v as follows:

$$\begin{aligned} e_t &= \mathbf{U}_a^\top \tanh\left(\mathbf{W}_a \mathbf{h}_t\right) \\ \alpha_t &= \frac{\exp\left(e_t\right)}{\sum_{i=1}^{T} \exp\left(e_i\right)} \\ v &= \Sigma_{i=1}^{T} \alpha_i h_i \end{aligned} \tag{4}$$

  where $\mathrm{W}_a \in \mathbb{R}^{2d \times 2d}$ and $\mathrm{U}_a \in \mathbb{R}^{2d \times 1}$ (we use d units for LSTM, thus h_t is a $2d$ dimensional vector) are parameters of attention layer. Using self-attention allow sentence to attend to itself, thus we can extract the most relevant information.

- **CNN-BiLSTM:**Similar as (Zhou et al., 2015), we first use CNN to extract a higher-level sequence representations, and then feed them into BiLSTM to obtain the final sentence representation. By combine CNN and BiLSTM, we are able to capture both lcal features of phrases and global informantion of sentence.

- **BiLSTM-CNN:** We also try to first use the BiLSTM layer as feature extrator and feed the hidden states to the CNN layer, which we call BiLSTM-CNN.

3.4 Ensemble Models

Classifier ensembles are a way of combining different models with the goal ofimroving overall peformance through enhanced decision making, which has been shown to achieve better results than single classifier. In this papaer, we explore 4 ensemble strategies to intergrate output (predict label or probability) from models introduced above and reach a decision.

- **Mean Probability:** Simply take an average of predictions from all the models and use it to make the final prediction.

- **Highest Confidence:** The class label that recieves vote with the highest probability is selected as the final prediction.

- **Majority Voting:** Each classifier votes for a single class label. The votes are summed and the label with majority votes (over 50%) wins. In case of tie, the ensemble falls back to the prediction by the model with highest peformance on validation set.
- **Meta-Classifier:** Use the individual classifier outputs along with training labels to train a second-level meta-classifier.The second meta-classifier servers to predict the final prediction. Meta-Classifier is also refered to as classifier stacking.

While the first three strategies use a simple fusion method to combine models, Meta-Classifier has parameters to tune, attempting to learn the collective knowledge represented by base classifier. As for estimator for meta-classfier, we experiment with a wide range of learning algorithms including not only the ones mentioned in Section3.2, such as random forest, GBDT and XGBoost. It turned out Gaussian Navie Bayes is the most competitive model, which will be the only meta classifer discusssed in next Section.

4 Experiments

4.1 Experimental setup

We use scikit-learn library[1] for implementation of the n-gram features based classifier and the ensemble meta-classifier. As for deep models, we implement them using Keras[2] library with Tensorflow backend. We used Adam (Kingma and Ba, 2014) method as the optimizer, setting the first momentum to be 0.9 , the second momentum 0.999 and the initial learning 0.001. The bacth size is 32. All hidden states of LSTMs, feature maps of CNNs and word embeddings have 300 dimensions. Word embeddings are fine tuned during training process. All the models are trained separately on traditional and simplified version's data set, and evaluated use accuracy metrics.Our code for all experiments is publicly available[3].

4.2 Contribution of single n-gram features

To find the most contributing individual n-gram features for discriminating Mandarin Chinese varieties. We run a number of experiments with the three classifiers using one single n-gram at a time, and the results are illustrated in Figure 2. As we can see, for data set of both simplified and traditional version, performances of 3 models all drop sharply when using a n-gram of larger size, on account of sparsity problem. The most contributing character level ngram is character tri-grama, to which character bi-gram is inferior yet close. Word uni-gram is the best in word level n-gram, but no better than character bi-gram or tri-gram. Among all 3 models, MNB outperforms LR and SVM, even though SVM has been the most preferred method for dialect identification. Besides, we can also see from the Figure 2, performances on data set of traditional version are slightly better than simplified version, which is persist with different models.

[1]https://scikit-learn.org/stable/

[2]https://github.com/keras-team/keras

[3]https://github.com/AlexYangLi/DMT

4.3 Combination of n-gram features

Table 3 shows the results of combining individual feature on each dataset. The performances of individual feature are also listed for direct comparison. As indicated from the table, feature combination does bring a performance gain. While 3 kinds of combination achieve a close score, MNB using combination of character bigram and trigram is the best, achiving accuracy score of 0.908 for simplfied version dataset and 0.9225 for traditional version.

4.4 Performance of deep learning models

To fully compare deep learning methods with machine learning methods for DMT task, 7 deep models are evaluated. Results are listed in Table 4. Among 7 deep models, BiLSTM stands out from others with accuracy of 0.9 and 0.9115. All models outperform LR and SVM, but are inferior to MNB, which shows MNB is a very strong classifier for discriminating variations of Mandarin Chinese.

4.5 Performance of ensemble models

We also try to achieve a better result by aggregating the outputs of models we have implemented. As presented in Table 5, no single ensemble strategy performs consistenly better than the others. The best choice for ensemble model is using MNB and BiLSTM as base classifier, and Mean Probability or Highest Confidence as fusion method. (When there are only 2 base classifiers, results of Mean Probability and Highest Confidenceare are always the same.)

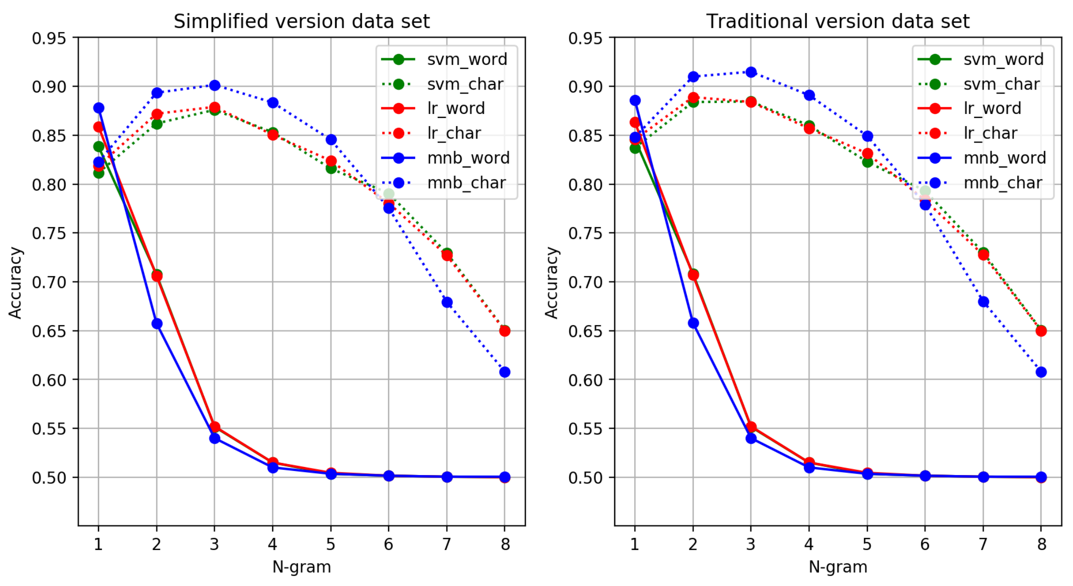

Figure 2: Accuracy scores for LR (red lines), SVM (green lines), MNB (blue lines) using character (solid lines) or word level n-gram of different sizes as input, both on data set of simplified (left) and traditional (right) version.

	Simplified			Traditional		
	LR	SVM	MNB	LR	SVM	MNB
Individual feature						
word uigram	0.859	0.8385	0.8785	0.8635	0.846	0.886
char uigram	0.8185	0.8115	0.8225	0.8455	0.837	0.848
char bigram	0.872	0.862	0.8935	0.889	0.884	0.91
char trigram	0.879	0.876	0.9015	0.884	0.8845	0.915
Combined feature						
char bigram+trigram	0.8865	0.883	**0.908**	0.896	0.8925	**0.9225**
char unigram+bigram+trigram	0.886	0.882	0.903	0.899	0.8965	0.917
char bigram+char trigram+word unigram	0.8875	0.8835	0.9055	0.899	0.894	0.92

Table 3: Accuracy scores for LR, SVM, MNB using individual or combine features as input, both on data set of simplified and traditional version.

	Simplfied	Traditional
CNN-based		
CNN	0.8965	0.909
DCNN	0.897	0.908
DPCNN	0.8925	0.907
RNN-based		
BiLSTM	**0.9**	**0.9115**
Self-attentive BILSTM	0.8915	0.902
CNN-RNN hybrid		
CNN-BiLSTM	0.8935	0.908
BiLSTM-CNN	0.895	0.9095

Table 4: Accuracy scores for deep models using word embeddings as input, both on data set of simplified and traditional version.

	simplfied			traditional		
	all ml*	all dl*	MNB + BiLSTM	all ml*	all dl*	MNB + BiLSTM
Mean Probability	0.9025	0.905	**0.913**	0.917	0.9215	**0.924**
Highest Confidence	0.908	0.9015	**0.913**	0.924	0.91	**0.924**
Majority Voting	0.888	0.906	-	0.8985	0.9195	-
Meta-Classifier	0.8915	0.9005	0.9065	0.906	0.9155	0.9215

Table 5: Accuracy scores for 4 ensemble strategies combining different base classifiers, both on data set of simplified and traditional version. "all ml" and "all dl" refer to combine all machine learning models and deep learning models respetively. All machine learning models use character bigram-trigram combination as input.

4.6 Results of shared task

To do.

5 Conclusion and feature work

To do.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Peng Chen, Zhongqian Sun, Lidong Bing, and Wei Yang. 2017a. Recurrent attention network on memory for aspect sentiment analysis. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 452–461. Association for Computational Linguistics.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017b. Enhanced lstm for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1657–1668. Association for Computational Linguistics.

Alina Maria Ciobanu and Liviu P. Dinu. 2016. A computational perspective on the romanian dialects. In *Proceedings of Language Resources and Evaluation (LREC)*.

Simon Clematide and Peter Makarov. 2017. Cluzh at vardial gdi 2017: Testing a variety of machine learning tools for the classification of swiss german dialects. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pages 170–177. Association for Computational Linguistics.

Çağrı Çöltekin, Taraka Rama, and Verena Blaschke. 2018. Tübingen-oslo team at the vardial 2018 evaluation campaign: An analysis of n-gram features in language variety identification. In *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018)*, pages 55–65. Association for Computational Linguistics.

Minwei Feng, Bing Xiang, Michael R. Glass, Lidan Wang, and Bowen Zhou. 2015. Applying deep learning to answer selection: A study and an open task. *CoRR*, abs/1508.01585.

Chu-Ren Huang and Lung-Hao Lee. 2008. Contrastive approach towards text source classification based on top-bag-of-word similarity. In *Proceedings of the 22nd Pacific Asia Conference on Language, Information and Computation*.

Rie Johnson and Tong Zhang. 2017. Deep pyramid convolutional neural networks for text categorization. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 562–570. Association for Computational Linguistics.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 655–665. Association for Computational Linguistics.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Tim Kreutz and Walter Daelemans. 2018. Exploring classifier combinations for language variety identification. In *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018)*, pages 191–198. Association for Computational Linguistics.

Zhouhan Lin, Minwei Feng, Cícero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. *CoRR*, abs/1703.03130.

Marco Lui and Paul Cook. 2013. Classifying english documents by national dialect. In *Proceedings of the Australasian Language Technology Association Workshop 2013 (ALTA 2013)*, pages 5–15.

Shervin Malmasi, Marcos Zampieri, Nikola Ljubešić, Preslav Nakov, Ahmed Ali, and Jörg Tiedemann. 2016. Discriminating between similar languages and arabic dialect identification: A report on the third dsl shared task. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial3)*, pages 1–14. The COLING 2016 Organizing Committee.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017a. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017b. Attention is all you need. *CoRR*, abs/1706.03762.

Fan Xu, Mingwen Wang, and Maoxi Li. 2016. Sentence-level dialects identification in the greater china region. *International Journal on Natural Language Computing (IJNLC)*, 5(6).

Marcos Zampieri and Binyam Gebrekidan Gebre. 2012. Automatic identification of language varieties: The case of portuguese. In *Proceedings of KONVENS*.

Marcos Zampieri, Shervin Malmasi, Nikola Ljubešić, Preslav Nakov, Ahmed Ali, Jörg Tiedemann, Yves Scherrer, and Noëmi Aepli. 2017. Findings of the vardial evaluation campaign 2017. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pages 1–15. Association for Computational Linguistics.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Ahmed Ali, Suwon Shon, James Glass, Yves Scherrer, Tanja Samardžić, Nikola Ljubešić, Jörg Tiedemann, Chris van der Lee, Stefan Grondelaers, Nelleke Oostdijk, Dirk Speelman, Antal van den Bosch, Ritesh Kumar, Bornini Lahiri, and Mayank Jain. 2018. Language identification and morphosyntactic tagging: The second vardial evaluation campaign. In *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018)*, pages 1–17. Association for Computational Linguistics.

Chunting Zhou, Chonglin Sun, Zhiyuan Liu, and Francis C. M. Lau. 2015. A C-LSTM neural network for text classification. *CoRR*, abs/1511.08630.