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, as it requires the ability to discriminate between varieties of one language. In this paper, we propose an ensemble based system, which combines 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 combination based Naive Bayes is a very strong model for identifying varieties of Mandarin Chinense. Through further ensemble of Navie Bayes and BiLSTM, our system (team: *itsalexyang*) achived an macro-averaged F1 score of 0.8530 and 0.8687 in two tracks.

1 Introduction

Dialect identification, which aims at distinguishing related languages or varieties of a specific language, is a special case of language identification. Accurate detection of dialects is an important step for many NLP piplines and applications, such as automatic speech recognition, machine translation and multilingual data acquisition. While there are effective solutions to language identification, dialect identification remains a tough problem to be tackled. As linguistic differences among related languages are less obvious than those among different languages, dialect identification is more subtle and complex, and therefore has become an attractive topic for many researchers in recent years.

Mandarin Chinese is a group of related 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. Mainland Chinese uses simplified Chinese characters, as opposed to the traditional Chinense characters used by Taiwanese. Take "*natural language processsing*" for example - its simplified character form adopted in Mainland China is "自然语言处理", while the traditional character form in Taiwan is "自然語言處理". Second, some vocabularies differ. Although some terms are mutually intelligible, they are preferred in one region. Table 1 lists some examples. Apart from character form and vocabularies, pronunciations, especially intonations, are also different. But we don't discuss this aspect, as it is irrelevant to the task.

Term	Mainland China	Taiwan
taxi	出租车	計程車
bicycle	自行车	腳踏車
software	软件	軟體
program	程序	程式
kindergarten	幼儿园	幼稚園

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

The DMT task, first introduced by VarDial evaluation campagin (Zampieri et al., 2019) this year, aims at determining whether a sentence belongs to news articles from Mainland China or from Taiwan. The organizers prepare two versions of the same corpus, traditional and simplified, and ask participants to predict the labels for text instances in both tracks. For that reason, character form can not be used to discriminate between these two language varieties. Mainstream approach to dialect identification is to regard it as a text classification task and use a linear support vector ma-

Proceedings of VarDial, pages 120–127

chine (SVM) with bag of n-gram features as input to slove it. However, we seek to find out what's the best classification algorithm for DMT task. Therefore, we experiment with serveral classical machine learning models trained on different word or character level n-gram features and feature combinatons. Besides, deep learning methods have currently achieve remakable sucess in many NLP tasks, including question answering, sentiment analysis, machine translation and natural language inference. To inverstigate how much deep neural neworks can help identify language varieties, we test 7 different deep learning models, including CNN based, RNN based and CNN-RNN hybrid models. Thorough performance comparison to machine learning models is also conducted. Finally, we explore different ways to ensemble the classifiers we discuss before.

2 Related Work

A number of works have devoted to differentiate between language varieties or related languages, especially since the series of VarDial evaluation campaigns (Zampieri et al., 2017, 2018, 2019). (Lui and Cook, 2013) studies on English dialect identification and presents serveral classification approaches to classify Australia, British and Caniadian English. (Zampieri and Gebre, 2012) utilizes a character n-gram and a word n-gram language model for automatic classificaton of two written varieties of Portuguese: European and Brazilian. (Ciobanu and Dinu, 2016) conducts an intial study on the dialects of Romanian and proposes using the orthographic and phonetic features of the words to build a dialect classifier. (Clematide and Makarov, 2017) uses 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 whether a document is in Flemish Dutch or Netherlandic Dutch. (Çöltekin et al., 2018) uses a unified SVM model based on character and word n-grams features with careful hyperparameter tuning for 4 language/dialect identification tasks.

Methods to discriminate between varieties of Mandarin Chinese haven't been well studied. (Huang and Lee, 2008) uses a top-bag-of-word similarity based contrastive approach to reflect distance among three varieties of Mandarin: Mainland China, Singapore and Taiwan. (Xu et al., 2016) deals 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. However, a thorough comparison of different algorithms and architectures has yet to be conducted.

3 Data and Methodology

3.1 Data

The DMT task is provided with labeled sentences from news published in Mainland China or in Taiwan (Chen et al., 1996; McEnery and Xiao, 2003). They are composed of 18770 instances for training set, 2000 for validation set and 2000 for test set. As shown in Table 2, the DMT dataset has a perfectly balanced class distribution. The avergae sentence lengths (in word level) of two varieties are almost the same. It's worth mentioning that the organizers have prepared two version of the same dataset: traditional and simplified version, which means we can't utilize character form feature to discriminate between these two language varieties. Since the sentences have been tokenized and punctuation has been removed from the texts, we don't apply any preprocessing on the dataset.

3.2 Traditional Machine Learning Models

Traditional machine learning models based on feature engineering are the most common methods for dialect identification. In this paper, we experiment with 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. The bag of n-gram features include word and character level n-grams with sizes ranging from 1 to a specific number. We conduct a set of experiments to fully explore the most contributing feature and feature combination for the DMT task, and the results are shown in next Section.

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: Statistics of dataset for each variety. Sentence lengths are calculated based on word-level tokens from training and validation set.

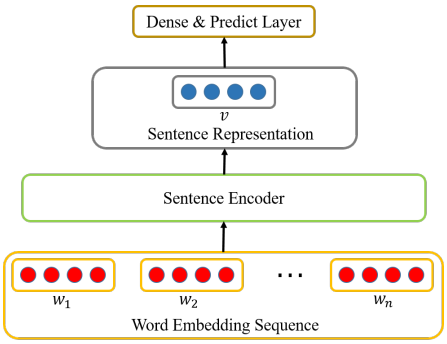

Figure 1: The overall framework of deep models.

3.3 Deep Learning Models

Deep neural networks (DNNs) are of growing interest for their capacity to learn text 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 examine a number of deep learning models based on a common framework to solve the DMT task. Figure 1 shows a high-level view of the framework. Vertically, the figure depicts 3 major components: (1) **Input Embedding Layer**. Suppose a sentence has n tokens, we use a pre-trained embedding method Word2vec (Mikolov et al., 2013) trained on training data to represent it in a sequence of word embeddings:

$$S = (\mathbf{w_1}, \mathbf{w_2}, \cdots \mathbf{w_n}) \tag{1}$$

where w_i is a vector representing a d dimentional word embedding for the i-th word in the sentence. S is thus a concatenation of all word embeddings. We do try using character embeddings and other pre-trained embedding methods 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**. The sentence encoder, specified by different deep learning models, processes the input word embedding sequence and outputs a high level sentence representation:

$$v = encode\,(S) \tag{2}$$

(3) **Output Layer**. After obtaining sentence vector, we feed it through one hidden dense layer with 256 units and a final predict dense layer:

$$\hat{y} = \sigma\left(\mathbf{W_p}\gamma\left(\mathbf{W_h}v + \mathbf{b_h}\right) + \mathbf{b_p}\right) \tag{3}$$

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, $\hat{\mathbf{y}} \in \mathbb{R}$ represents the predicted score for postive class. During training process, we minimize the binary cross-entropy loss defined as follow:

$$L = -\frac{1}{N}\sum_{i=1}^{N}\left(y_i \log \hat{y}_i + (1 - y_i) \log\left(1 - \hat{y}_i\right)\right) \tag{4}$$

where y_i is the ground-truth.

We examine 7 different deep learning models to encode sentences into fixed-size vectors, including CNN-based, RNN-based and CNN-RNN hybird neural networks.

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

$$c_i = f\left(\mathbf{W_c} \cdot \mathbf{w}_{i:i+h-1} + b\right) \tag{5}$$

  After applying this filter to each possible window of words in a sentence, a feature map can be produced. In this paper, we use 300 filters with window sizes 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 features 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. Through a k-Max pooling operation, a feature graph over the sentence can be induced, which explicitly captures 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 connection) interleaved with pooling layers with stride 2 for downsampling. The 2-stride downsampling reduces the size of the internal represenation of each text by half, enabling efficient representation of long-range association in the text. The shortcut connection ensures training of deep networks. DPCNN has been shown powerful in many text classification 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, so 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:

$$\begin{aligned} \overrightarrow{h_t} &= \overrightarrow{\text{LSTM}}\left(\mathbf{w_1}, \mathbf{w_2}, \ldots, \mathbf{w_t}\right) \\ \overleftarrow{h_t} &= \overleftarrow{\text{LSTM}}\left(\mathbf{w_n}, \mathbf{w_{n-1}}, \ldots, \mathbf{w_t}\right) \\ h_t &= \left[\overrightarrow{h_t}, \overleftarrow{h_t}\right] \end{aligned} \tag{6}$$

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

- **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., 2017). 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 states 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:

$$\begin{aligned} e_t &= \mathbf{U}_a^\top \tanh\left(\mathbf{W}_a \mathbf{h}_t\right) \\ \alpha_t &= \frac{\exp\left(e_t\right)}{\sum_{i=1}^{T} \exp\left(e_i\right)} \\ v &= \Sigma_{i=1}^{T} \alpha_i h_i \end{aligned} \tag{7}$$

  where $\mathrm{W}_a \in \mathbb{R}^{2d \times 2d}$ and $\mathrm{U}_a \in \mathbb{R}^{2d \times 1}$ are parameters of the attention layer (we use d units for LSTM, thus h_t being a $2d$ dimensional vector). Using self-attention allows a sentence to attend to itself, therefore 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 from word embedding sequences, and then feed them into BiLSTM to obtain final sentence representation. By combing CNN and BiLSTM, we are able to capture both local features of phrases and global informantion of sentences.

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

3.4 Ensemble Models

Classifier ensemble is a way of combining different models with the goal of improving overall peformance through enhanced decision making, which has been shown to achieve better results than a single classifier. In this paper, we explore 4 ensemble strategies to intergrate outputs (predicted labels or probabilities) from models introduced above and reach a final 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 a tie, the ensemble result 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 then predicts 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 train, which attempts to learn the collective knowledge represented by base classifiers. As for choosing estimators for meta-classfier, we test with a wide range of learning algorithms including not only the ones mentioned in Section 3.2, but also random forest, GBDT, XGBoost and so on. It turns 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 the implementation of the n-gram features based models and the ensemble meta-classifier. As for deep learning 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 models are trained separately on dataset of traditional and simplified version, and evaluated using macro-weighted f1 score. Our code for all experiments is publicly available[3].

4.2 Contribution of Single N-gram Feature

To find the most contributing individual n-gram feature for discriminating between 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 shown in Figure 2. In terms of n-gram features, for dataset of both simplified and traditional version, performances of 3 models all drop sharply as n-gram size increases, especially for word level n-grams. The most contributing character level ngram is character trigram, which is slightly better than character bigram. Word unigram is the best among word level n-grams, but no better than character bigram or trigram. As for the 3 models, although SVM has been the most preferred method for dialect identification, in our experiment, MNB outperforms LR and SVM. Lastly, for all models, performance on the traditional version dataset is slightly better than that on the simplified version dataset.

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. MNB with character bigram and trigram combination achieves the highest macro-weighted f1 scores, 0.9080 for the simplified version and 0.9225 for the traditional version.

4.4 Performance of Deep Learning Models

To fully compare deep learning methods with machine learning methods for the DMT task,we evaluate 7 deep learning models. Results are listed in Table 4. Among these models, BiLSTM stands out from the others with macro-weighted f1 scores of 0.9000 and 0.9115. All deep learning models outperform LR and SVM, but are inferior to MNB, which shows again MNB is a very strong classifier for discriminating between varieties of Mandarin Chinese.

4.5 Performance of Ensemble Models

We also try to achieve a better result by aggregating outputs of the 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 Confidence are always the same.)

4.6 Results of Shared Task

We submit 3 systems for the evalution of test set: MNB, BiLSTM and their ensemble. The official results of our submissons show the same pattern observed on the validation set (see Table 6). MNB performs better than BiLSTM, especially for the

[1] https://scikit-learn.org/stable/
[2] https://github.com/keras-team/keras
[3] https://github.com/AlexYangLi/DMT

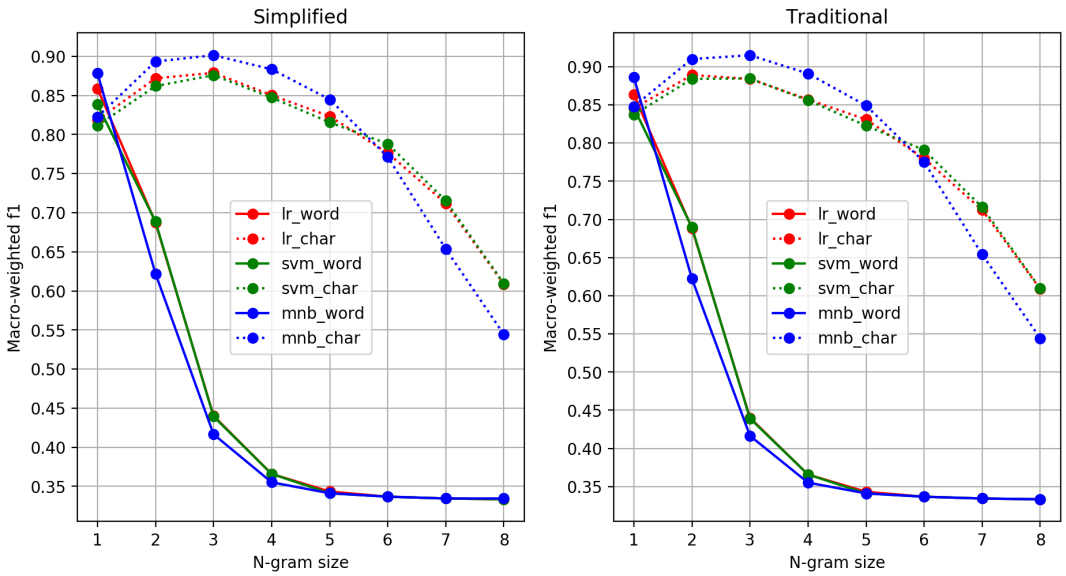

Figure 2: Macro-weighted f1 scores of LR (red lines), SVM (green lines), MNB (blue lines) using character (dotted lines) or word level (solid lines) n-gram of different sizes as input, both on dataset of simplified (left) and traditional (right) version.

Feature	**Simplified**			**Traditional**		
	LR	SVM	MNB	LR	SVM	MNB
Individual feature						
word uigram	0.8590	0.8384	0.8784	0.8634	0.8460	0.8860
char bigram	0.8720	0.8620	0.8935	0.8890	0.8840	0.9100
char trigram	0.8790	0.8760	0.9015	0.8840	0.8845	0.9150
char 4gram	0.8504	0.8474	0.8835	0.8570	0.8559	0.8910
Combined feature						
char bigram+trigram	0.8865	0.8830	**0.9080**	0.8960	0.8925	**0.9225**
char bigram+trigram+4gram	0.8880	0.8835	0.9030	0.8945	0.8920	0.9170
char bigram+char trigram+word unigram	0.8875	0.8835	0.9055	0.8990	0.8940	0.9200

Table 3: Macro-weighted f1 scores of LR, SVM, MNB using individual or combined features as input, both on dataset of simplified and traditional version.

Model	**Simplfied**	**Traditional**
CNN-based		
CNN	0.8964	0.9090
DCNN	0.8970	0.9080
DPCNN	0.8925	0.9070
RNN-based		
BiLSTM	**0.9000**	**0.9115**
Self-attentive BiLSTM	0.8915	0.9020
CNN-RNN hybrid		
CNN-BiLSTM	0.8935	0.9080
BiLSTM-CNN	0.8950	0.9095

Table 4: Macro-weighted f1 scores of deep learning models using word embeddings as input, both on dataset of simplified and traditional version.

Ensemble Strategy	Simplfied			Traditional		
	all ML*	all DL*	MNB + BiLSTM	all ML*	all DL*	MNB + BiLSTM
Mean Probability	0.9025	0.9050	**0.9130**	0.9170	0.9215	**0.9240**
Highest Confidence	0.9080	0.9015	**0.9130**	0.9225	0.9100	**0.9240**
Majority Voting	0.8880	0.9060	-	0.8985	0.9195	-
Meta-Classifier	0.8915	0.9025	0.9050	0.906	0.9130	0.9215

Table 5: Macro-weighted f1 scores of 4 ensemble strategies combining different base classifiers, both on dataset of simplified and traditional version. "all ML" and "all DL" refer to combine all machine learning models and deep learning models respectively. All machine learning models use character bigram-trigram combination as input.

Submission	Simplified	Traditional
BiLSTM	0.8118	0.8450
MNB	0.8499	0.8650
MNB + BiLSTM	**0.8530**	**0.8687**

Table 6: Macro-weighted f1 scores of 3 submissions on test sets (team: *itsalexyang*).

simplified version of test data. In addition, the MNB-BiLSTM ensemble achieves a higher score than a single model for both versions of test data. Overall, our models' performance is consistently lower on the test set than on the validation set. We believe tuning parameters with k-fold cross validation or applying other overfitting prevention strategies may help yield better results on unseen data.

5 Conclusion

In this paper, we describes our submission for the DMT task. Our experiments show that multinomial Naive Bayes is a very strong model for discrinating between Mandarin varieties, which works better than the most commonly used SVM and popular deep learning models. For MNB, character trigram is the most contributing feature. Further performance gain can be achieved by combining character trigram and bigram feature. We also explore different ways to ensemble models, and find that average ensemble (or highest confidence ensemble) of MNB and BiLSTM is the best model for the DMT task.

In future work, we would like to apply our model to deal with more varieties of Mandarin Chinese (e.g. Hong Kong, Taiwan, Macao, Singapore and Malaysia) to examine its effectiveness.

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.

Keh-Jiann Chen, Chu-Ren Huang, Li-Ping Chang, and Hui-Li Hsu. 1996. SINICA CORPUS : Design Methodology for Balanced Corpora. In *Language, Information and Computation : Selected Papers from the 11th Pacific Asia Conference on Language, Information and Computation : 20-22 December 1996, Seoul*, pages 167–176, Seoul, Korea. Kyung Hee University.

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.

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.

A. M. McEnery and R. Z. Xiao. 2003. The lancaster corpus of mandarin chinese.

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, Lukasz Kaiser, and Illia Polosukhin. 2017. 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.

Marcos Zampieri, Shervin Malmasi, Yves Scherrer, Tanja Samardžić, Francis Tyers, Miikka Silfverberg, Natalia Klyueva, Tung-Le Pan, Chu-Ren Huang, Radu Tudor Ionescu, Andrei Butnaru, and Tommi Jauhiainen. 2019. A Report on the Third VarDial Evaluation Campaign. In *Proceedings of the Sixth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*. 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.