Combining BERT with Static Word Embeddings for Categorizing Social Media

Israa Alghanmi, Luis Espinosa-Anke, Steven Schockaert
Cardiff University, UK
{alghanmiia,espinosa-ankel,schockaerts1}@cardiff.ac.uk

Abstract

Pre-trained neural language models (LMs) have achieved impressive results in various natural language processing tasks, across different languages. Surprisingly, this extends to the social media genre, despite the fact that social media often has very different characteristics from the language that LMs have seen during training. A particularly striking example is the performance of AraBERT, an LM for the Arabic language, which is successful in categorizing social media posts in Arabic dialects, despite only having been trained on Modern Standard Arabic. Our hypothesis in this paper is that the performance of LMs for social media can nonetheless be improved by incorporating static word vectors that have been specifically trained on social media. We show that a simple method for incorporating such word vectors is indeed successful in several Arabic and English benchmarks. Curiously, however, we also find that similar improvements are possible with word vectors that have been trained on traditional text sources (e.g. Wikipedia).

1 Introduction

Social media has become an important source of information across numerous disciplines (Jaffali et al., 2020). For instance, it allows extracting and analyzing people's opinions, emotions and attitudes towards particular subjects, in a way which is difficult to achieve using other information sources. However, social media posts tend to be short and often contain abbreviations, slang words, misspellings, emoticons and dialect (Baly et al., 2017). For language models (LMs) such as BERT (Devlin et al., 2019), which have been primarily trained on Wikipedia, this poses a number of clear challenges. In the case of Arabic, the challenge is even greater, since social media posts are mostly written in regional dialects, which can be different from the language that is found in resources such as Wikipedia (Alali et al., 2019). In particular, the Arabic language can be divided into Classical Arabic, Modern Standard Arabic, and Dialectal Arabic (Alotaibi et al., 2019). The latter differs between Arabic countries, and sometimes among regions and cities. Social Media acts as the primary source where Arabic dialects appear as written text, due to the informality of these platforms.

Similar as for English, the best results in many Arabic NLP tasks are currently obtained with LMs. In particular, the AraBERT model (Antoun et al., 2020) has achieved state-of-the-art results in sentiment analysis, named entity recognition and question answering, among others. However, AraBERT was trained on Wikipedia and news stories. It has thus not seen the Arabic dialects in which most social media posts are written. Surprisingly, however, Antoun et al. (2020) found that AraBERT is nonetheless able to outperform other methods on social media tasks. This includes methods that use the AraVec (Soliman et al., 2017) embeddings, which are word2vec vectors trained on Twitter, and have a wide coverage of dialect words.

Our hypothesis is that AraBERT and AraVec have complementary strengths, and that better results can thus be obtained by combining these two resources. Similarly, for English tasks, we would expect that the performance of BERT on social media can be improved by incorporating word embeddings that have been trained on social media. However, for English we would expect to see a smaller effect, since compared to Arabic, the vocabulary of English social media is more similar to the vocabulary in traditional sources. To test these hypotheses, we propose and evaluate a simple classifier which combines language models with static word embeddings. Our main findings are that incorporating word vectors can indeed boost performance. Surprisingly, this even holds for word embeddings that have been trained on standard sources.

Proceedings of the 2020 EMNLP Workshop W-NUT: The Sixth Workshop on Noisy User-generated Text, pages 28–33

2 Related Work

While there is a large literature on NLP for social media, more efforts that focus on the Arabic language are needed. A notable work is Heikal et al. (2018), which developed a CNN and LSTM ensemble model for Arabic sentiment analysis. They used AraVec pre-trained word embeddings for the word embedding representation. Recently, Kaibi et al. (2020) proposed an approach that relies on the concatenation of pre-trained AraVec and fastText vectors. However, the best results on most datasets are currently achieved by fine-tuning AraBERT (Antoun et al., 2020), as already mentioned in the introduction. For the English language, Nguyen et al. (2020) recently introduced BERTweet, a BERT-based language model that was trained on a large corpus of English tweets. Their experiments show that utilising this model led to improved results on different tasks involving Twitter posts, such as named entity recognition, part-of-speech tagging and text classification.

In this work, we investigate the effectiveness of combining pre-trained language models with static word embeddings. For earlier language models, most notably ELMo (Peters et al., 2018), it was common practice to combine contextual embeddings, predicted by the language model, with static word embeddings. However, the introduction of BERT has essentially eliminated the need for static word vectors in standard settings. On the other hand, several authors have shown that it can be beneficial to incorporate entity vectors with BERT, allowing the model to exploit factual or commonsense knowledge from structured sources (Lin et al., 2019; Poerner et al., 2019).

3 Proposed Approach

There are various ways in which BERT-based models can be combined with static word vectors. Note, however, that we cannot simply concatenate the contextualised word vectors predicted by BERT with the corresponding static word vectors, due to the fact that the tokenization strategy used by BERT means that many words are split into two or more word-piece tokens. One possible solution, adopted by Zhang et al. (2020) in a different setting, is to combine the word-piece tokens from the same word into a single vector, using a convolutional or recurrent neural network. The resulting word-level vector can then be concatenated with the corresponding static word vector. However, without a large training set, there is a risk that the representations predicted by BERT are degraded by this aggregation step. As a simpler solution, we instead combine representations obtained from BERT and from the static word vectors at sentence level. In particular, to obtain a sentence vector from the fine-tuned BERT model, we simply take the average of the predicted contextualised vectors. To obtain a sentence vector from the static word embeddings, we use either a Convolutional Neural Network (CNN) or a Long Short Term Memory network (LSTM). After concatenating the two types of sentence vectors, we apply dropout, followed by a softmax classification layer. A diagram illustrating the model is shown in Figure 1. Rather than jointly training the combined model, we first fine-tune the BERT model on its own. After this fine-tuning step, we freeze the BERT model and train the CNN and combined classification layer. We found this strategy to be more robust against over-fitting.

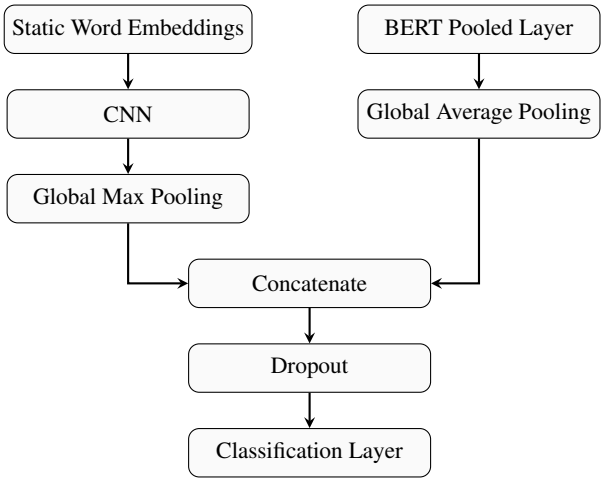

Figure 1: The CNN based proposed architecture

4 Experimental Results

We experimentally analyze the benefit of incorporating static word vectors, in both Arabic and English. For Arabic, we used the following datasets:

AJGT Arabic Jordanian General Tweets (Alomari et al., 2017) is a sentiment classification dataset. It consists of 1,800 tweets annotated with positive and negative labels.

SemEval-2017 Task 4 (Rosenthal et al., 2017) We use the Arabic subtask A dataset from SemEval 2017 Task 4. This dataset contains 10,126 tweets, annotated with negative, neutral, and positive labels.

L-HSAB A dataset for hate speech and abusive language in Arabic Levantine dialect (Mulki et al., 2019). It consists of 5846 tweets, which are annotated as normal, abusive or hate.

ArsenTD-Lev An Arabic Levantine dataset for sentiment analysis that discusses multiple topics (Baly et al., 2019). It contains 4000 tweets, labelled as a very negative, negative, neutral, positive, or very positive.

For English, we used the following datasets:

Irony Detection We use Semeval-2018 Task 3 dataset (Van Hee et al., 2018), containing 4,618 tweets that are annotated as ironic or not.

Semeval-2019 Task 6: OffensEval An offensive language identification dataset for social media (Zampieri et al., 2019). It consist of 14,100 tweets that are labeled as offensive or not.

Stance Detection We use the climate change Semeval-2016 Task 6 subtask A dataset (Mohammad et al., 2016). It contains 564 tweets, annotated as in favour, against or neutral towards the target.

Hate Speech Detection We utilize the English SemEval-2019 Task 5 dataset (Basile et al., 2019). It contains 13,000 tweets labeled as hateful or not.

For datasets without standard splits, we randomly split the data into 80% for training and 20% for testing. We removed emojis, hashtag signs (#), numbers, special characters and punctuation. We replace user mentions with [user], URLs with [link], and email addresses with [email], written in Arabic or English depending on the language of the dataset. For Arabic tweets, we also removed diacritics, elongation and English letters, besides shallow normalization. For English, we removed non-ASCII characters, convert emoticons to text, and common contractions to their full form.

Word Embeddings. For the Arabic dataset, we use AraVec word vectors (Soliman et al., 2017), which are based on the word2vec model (Mikolov et al., 2013). Pre-trained models are available for two Arabic content domains: Wikipedia and Twitter. For each domain, Skip-gram and CBOW versions with 100 and 300 dimension vector sizes were provided. In our experiments, for both domains, we have used the 300-dimensional CBOW vectors (**AraVec-twi** and **AraVec-wiki**). For English, we have used GloVe vectors provided by (Pennington et al., 2014). We have used the 100 dimensional word vectors that have been pre-trained on Twitter data (**GloVe-twi**), as well as the 100-dimensional GloVe vectors that have been trained on Wikipedia (**GloVe-wiki**).

Language Models. We use the pre-trained AraBERTv0.1 model[1] (Antoun et al., 2020) for Arabic and the $\text{BERT}_{\text{base}}$ uncased (Devlin et al., 2018) model for English.

Baselines and Methodology. As a baseline, we show the performance of a standard CNN, using only the static word vectors as input. We use 100 convolutional filters, a kernel size of 3, and a ReLU activation function. A global max-pooling layer follows the convolution layer. A dropout layer with a 0.5 drop rate is applied to the max-pooled output to avoid over-fitting. We use SGD with a batch size of 16 for 15 epochs with early stopping callback.

We also show results for BERT and AraBERT alone. We followed the BERT TensorFlow implementation for sequence classification provided by Hugging Face (Wolf et al., 2019). Both AraBERT and the English $\text{BERT}_{\text{base}}$ pre-trained language models share the same architecture, which consists of 12 layers. We utilize the Adamax optimizer and batch size of 8. The hyper-parameters search for the fine-tuning process involves the number of epochs (3 to 6) and the learning rates [2e-5, 5e-5]. We chose the best performing hyper-parameters based on a validation split. We use the standard validation split for datasets where one is provided, and use 20% of the training data as validation otherwise. We fine-tune BERT on the whole training data once the best hyper-parameters are chosen.

For the CNN variant of our proposed hybrid approach, we use the same configuration as for the CNN baseline, i.e. we use a convolution layer with a filter size of 100, a kernel size of 3, and the ReLU activation function, followed by global max-pooling. For the LSTM variant, we use 100-dimensional units. In both variants, the dropout rate is set to 0.5, and we use SGD with a batch size of 16, for 15 epochs, with the usage of early stopping callback.

[1] There are two versions of AraBERT, called AraBERTv0.1 and AraBERTv1. The only difference is that a Farasa Segmenter is used for the latter.

Model	Embeddings	AJGT	SemEval-2017	L-HSAB	ArsenTD-Lev
CNN	AraVec-twi	89.9	55.6	60.9	47.5
AraBERT	-	93.3	63.1	71.0	51.8
CNN + AraBERT	AraVec-twi	**93.4** (93.0, 93.9)	64.1 (63.5, 64.4)	**72.1** (71.7, 72.3)	49.2 (47.1, 51.5)
LSTM + AraBERT	AraVec-twi	93.1 (92.8, 93.6)	63.7 (63.3, 64.1)	72.0 (71.5, 72.5)	**52.2** (51.7, 52.7)
CNN + AraBERT	AraVec-wiki	**93.4** (92.8, 93.6)	**64.3** (63.9, 64.5)	71.8 (71.6, 72.1)	50.9 (48.0, 53.1)
LSTM + AraBERT	AraVec-wiki	93.1 (92.8, 93.6)	63.7 (63.4, 63.9)	71.9 (71.5, 72.5)	51.9(51.5, 52.5)

Table 1: F1 scores (%) for Arabic datasets. We report the average result from five runs for CNN, CNN+AraBERT and LSTM+AraBERT, as well as the minimum and maximum results between parentheses. The fine-tuned AraBERT model is fixed across all variants.

Model	Embeddings	Irony	OffensEval	Hate	Stance
CNN	GloVe-twi	57.2	75.1	47.0	29.0
BERT	-	67.3	78.5	**50.6**	52.9
CNN + BERT	GloVe-twi	**68.4** (67.4, 69.7)	79.4 (79.2, 79.7)	48.1 (47.6, 48.5)	54.3 (52.9, 56.0)
LSTM+ BERT	GloVe-twi	68.3 (67.6, 68.9)	79.5 (79.2, 79.8)	47.7 (47.5, 47.8)	54.1 (53.1, 55.5)
CNN + BERT	GloVe-wiki	67.7 (66.5, 68.7)	79.4 (79.0, 79.6)	48.1 (47.8, 48.3)	**54.6** (53.6, 55.5)
LSTM+ BERT	GloVe-wiki	68.3 (67.6, 68.9)	**79.6** (79.2, 79.8)	47.7 (47.5, 47.8)	54.1 (53.1, 55.5)

Table 2: F1 scores (%) for English datasets. We report the average result from five runs for CNN, CNN+BERT and LSTM+BERT, as well as the minimum and maximum results between parentheses. The fine-tuned BERT model is fixed across all variants.

Results. Table 1 summarizes the performance of the baseline models and the proposed strategy for the Arabic language, while Table 2 shows the results for English. The results for AraBERT and BERT are the best results that were obtained over three runs. We then fix this model and combine it with the CNN and LSTM models. The results of these combined models (and the CNN baseline) are averaged over 5 runs. We use this approach since the focus is on assessing whether the performance of BERT and AraBERT can be improved.

Overall, the proposed combined model improves the results across almost all datasets, with the CNN and LSTM variants performing broadly similarly. One exception for Arabic is the **ArsenTD-Lev** dataset, where the LSTM variant performs substantially better than the CNN variant. and the English **Hate** dataset, where neither of the two variants outperforms the fine-tuned BERT model. The underperformance on the **Hate** dataset is likely related to over-fitting, as there is a clear mismatch between training and test data in this dataset (e.g. in terms of annotation strategy and average tweet length). The most surprising finding is that the AraVec-twi and AraVec-wiki word embeddings achieve comparable performance for Arabic, and similarly, the GloVe-twi and GloVe-wiki embeddings achieve comparable performance for English. This suggests that the main improvements are not due to the fact that the word embeddings are specialized towards the social media genre, but rather because they capture complementary facets of word meaning. We conjecture that word vectors can, in particular, provide valuable complementary information for rare words. Schick and Schütze (2020) found that BERT struggles with rare words and we can indeed expect social media texts to contain a larger proportion of rare words than documents in other genres.

5 Conclusions

In this paper, we have presented a simple approach to combine static word embeddings with BERT-based language models. Intuitively, the reason why this hybrid approach can outperform the BERT-based models themselves is because the latter were not trained on Wikipedia. The alternative solution would be to train language models on a relevant social media corpus, as in the BERTweet model (Nguyen et al., 2020). While such a strategy is likely to lead to a better overall performance, in principle, this is not always feasible in practice. For instance, using static word vectors could play an important role in dealing with emerging terms, such as trending hashtags, as continuously updating language models (for many different languages) would be too expensive. Similarly, incorporating static word vectors seems to be a promising strategy for improving language models for low-resource languages, as specialized language models (e.g. trained on social media) are unlikely to become available for such languages.

References

Muath Alali, Nurfadhlina Mohd Sharef, Masrah Azrifah Azmi Murad, Hazlina Hamdan, and Nor Azura Husin. 2019. Narrow convolutional neural network for arabic dialects polarity classification. *IEEE Access*, 7:96272–96283.

Khaled Mohammad Alomari, Hatem M ElSherif, and Khaled Shaalan. 2017. Arabic tweets sentimental analysis using machine learning. In *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, pages 602–610. Springer.

Shoayee Alotaibi, Rashid Mehmood, and Iyad Katib. 2019. Sentiment analysis of arabic tweets in smart cities: A review of saudi dialect. In *2019 Fourth International Conference on Fog and Mobile Edge Computing (FMEC)*, pages 330–335. IEEE.

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. Arabert: Transformer-based model for arabic language understanding.

Ramy Baly, Gilbert Badaro, Georges El-Khoury, Rawan Moukalled, Rita Aoun, Hazem Hajj, Wassim El-Hajj, Nizar Habash, and Khaled Shaban. 2017. A characterization study of arabic twitter data with a benchmarking for state-of-the-art opinion mining models. In *Proceedings of the third Arabic natural language processing workshop*, pages 110–118.

Ramy Baly, Alaa Khaddaj, Hazem Hajj, Wassim El-Hajj, and Khaled Bashir Shaban. 2019. Arsentd-lev: A multi-topic corpus for target-based sentiment analysis in arabic levantine tweets. *arXiv preprint arXiv:1906.01830*.

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Nozza Debora, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, Manuela Sanguinetti, et al. 2019. Semeval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter. In *13th International Workshop on Semantic Evaluation*, pages 54–63. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186.

Maha Heikal, Marwan Torki, and Nagwa El-Makky. 2018. Sentiment analysis of arabic tweets using deep learning. *Procedia Computer Science*, 142:114–122.

Soufien Jaffali, Salma Jamoussi, Nesrine Khelifi, and Abdelmajid Ben Hamadou. 2020. Survey on social networks data analysis. In *International Conference on Innovations for Community Services*, pages 100–119. Springer.

Ibrahim Kaibi, Hassan Satori, et al. 2020. Sentiment analysis approach based on combination of word embedding techniques. In *Embedded Systems and Artificial Intelligence*, pages 805–813. Springer.

Bill Yuchen Lin et al. 2019. KagNet: Knowledge-aware graph networks for commonsense reasoning. In *EMNLP*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Saif Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. 2016. Semeval-2016 task 6: Detecting stance in tweets. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 31–41.

Hala Mulki, Hatem Haddad, Chedi Bechikh Ali, and Halima Alshabani. 2019. L-hsab: A levantine twitter dataset for hate speech and abusive language. In *Proceedings of the Third Workshop on Abusive Language Online*, pages 111–118.

Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. 2020. Bertweet: A pre-trained language model for english tweets. *arXiv preprint arXiv:2005.10200*.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2227–2237.

Nina Poerner, Ulli Waltinger, and Hinrich Schütze. 2019. E-bert: Efficient-yet-effective entity embeddings for bert.

Sara Rosenthal, Noura Farra, and Preslav Nakov. 2017. SemEval-2017 task 4: Sentiment analysis in Twitter. In *Proceedings of the 11th International Workshop on Semantic Evaluation*, SemEval '17, Vancouver, Canada. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2020. Rare words: A major problem for contextualized embeddings and how to fix it by attentive mimicking. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 8766–8774.

Abu Bakr Soliman, Kareem Eissa, and Samhaa R El-Beltagy. 2017. Aravec: A set of arabic word embedding models for use in arabic nlp. *Procedia Computer Science*, 117:256–265.

Cynthia Van Hee, Els Lefever, and Véronique Hoste. 2018. Semeval-2018 task 3: Irony detection in english tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 39–50.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. Semeval-2019 task 6: Identifying and categorizing offensive language in social media (offenseval). *arXiv preprint arXiv:1903.08983*.

Zhuosheng Zhang, Yuwei Wu, Hai Zhao, Zuchao Li, Shuailiang Zhang, Xi Zhou, and Xiang Zhou. 2020. Semantics-aware BERT for language understanding. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 9628–9635.