Modelling the Combination of Generic and Target Domain Embeddings in a Convolutional Neural Network for Sentence Classification

Nut Limsopatham and **Nigel Collier**
Language Technology Lab
Department of Theoretical and Applied Linguistics
University of Cambridge
Cambridge, UK
{nl347,nhc30}@cam.ac.uk

Abstract

Word embeddings have been successfully exploited in systems for NLP tasks, such as parsing and text classification. It is intuitive that word embeddings created from a larger corpus would provide a better coverage of vocabulary. Meanwhile, word embeddings trained on a corpus related to the given task or target domain would more effectively represent the semantics of terms. However, in some emerging domains (e.g. bio-surveillance using social media data), it may be difficult to find a domain corpus that is large enough for creating effective word embeddings. To deal with this problem, we propose novel approaches that use both word embeddings created from generic and target domain corpora. Our experimental results on sentence classification tasks show that our approaches significantly improve the performance of an existing convolutional neural network that achieved state-of-the-art performances on several text classification tasks.

1 Introduction

Word embeddings (i.e. distributed vector representation) represent words using dense, low-dimensional and real-valued vectors, where each dimension represents a latent feature of the word (Turian et al., 2010; Mikolov et al., 2013; Pennington et al., 2014). It has been empirically shown that word embeddings could capture semantic and syntactic similarities between words (Turian et al., 2010; Mikolov et al., 2013; Pennington et al., 2014; Levy and Goldberg, 2014). Importantly, word embeddings have been effectively used for several NLP tasks (Turian et al., 2010; Collobert et al., 2011; Segura-Bedmar et al., 2015; Limsopatham and Collier, 2015a; Limsopatham and Collier, 2015b; Muneeb et al., 2015). For example, Turian et al. (2010) used word embeddings as input features for several NLP systems, including a traditional chunking system based on conditional random fields (CRFs) (Lafferty et al., 2001). Collobert et al. (2011) used word embeddings as inputs of a multilayer neural network for part-of-speech tagging, chunking, named entity recognition and semantic role labelling. Limsopatham and Collier (2016) leveraged semantics from word embeddings when identifying medical concepts mentioned in social media messages. Kim (2014) showed that using pre-built word embeddings, induced from 100 billion words of Google News using word2vec (Mikolov et al., 2013), as inputs of a simple convolutional neural network (CNN) could achieve state-of-the-art performances on several sentence classification tasks, such as classification of positive and negative reviews of movies (Pang and Lee, 2005) and consumer products, e.g. cameras (Hu and Liu, 2004).

The quality of word embeddings (e.g. the ability to capture semantics of words) highly depends on the corpus from which they are induced (Pennington et al., 2014). For instance, when induced from a generic corpus, such as Google News, the vector representation of 'tissue' would be similar to the vectors of 'paper' and 'toilet'. However, when induced from medical corpora, such as PubMed[1] or BioMed Central[2], the vector of 'tissue' would be more similar to those of 'cell' and 'organ'. Hence, word embeddings induced from the corpus related to the task or target domain are likely to be more useful. Meanwhile, it is intuitive that the more training documents used, the more likely that more vocabulary is covered. Recent studies (e.g. (Faruqui et al., 2015; Xu et al., 2014; Yu and Dredze, 2014)) have attempted to improve the quality of word embeddings by

[1] https://www.ncbi.nlm.nih.gov/pubmed/
[2] https://www.biomedcentral.com/

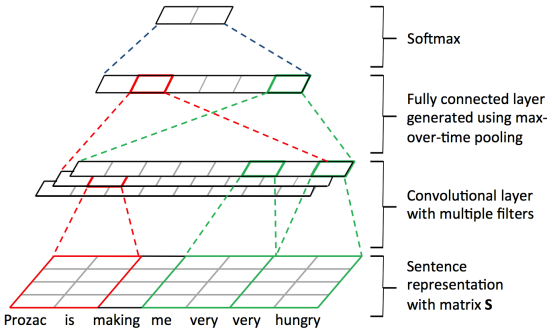

Figure 1: CNN for sentence classification.

enhancing the learning algorithm or injecting an existing knowledge-base, e.g. WordNet (Miller, 1995) or UMLS semantic network[3]. Pennington et al. (2014) incorporated aggregated global word co-occurrence statistics from the corpus when inducing word embeddings. Xu et al. (2014) and Yu and Dredze (2014) exploited semantic knowledge to improve the semantic representation of word embeddings. Nevertheless, in some emerging domains, e.g. detecting adverse drug reactions (ADR) reported in social media, existing knowledge resources or corpora may not be large enough for creating effective embeddings.

In this work, we investigate novel approaches to incorporate both generic and target domain embeddings in CNN for sentence classification. We hypothesise that using both generic and target domain embeddings further improves the performance of CNN, since it can benefit from both the good coverage of vocabulary from the generic embedding, and the effective semantic representation of the target domain embedding. This would enable CNN to perform effectively without requiring new target domain embeddings induced from a large amount of domain documents specifically related to individual tasks. We thoroughly evaluate our proposed approaches using an ADR tweet classification task (Ginn et al., 2014). In addition, to show that our approaches are effective for different target domains, we also evaluate them using a movie review classification task (Pang and Lee, 2005). Our experimental results show that our approaches significantly improve the performance in term of accuracy over an existing strong baseline that uses only either the generic or the target domain embeddings.

2 CNN for Sentence Classification

CNN has been used to model sentences in different NLP tasks, such as sentence classification and sentence matching (Collobert and Weston, 2008; Kim, 2014; Kalchbrenner et al., 2014; Hu et al., 2014). In this work, we adapt the CNN model of Kim (2014) to exploit both generic and target domain word embeddings, because of its simplicity and effectiveness. The model architecture of Kim (2014) is shown in Figure 1. In particular, for a given input sentence of length n words (padded where necessary), we create a sentence matrix $\mathbf{S} \in \mathbb{R}^{d \times n}$, where each column is the d-dimensional vector (i.e. embedding) $\mathbf{x}_i \in \mathbb{R}^d$ of each word in the sentence:

$$\mathbf{S} = \begin{bmatrix} | & | & | & | \\ \mathbf{x}_1 & \mathbf{x}_2 & ... & \mathbf{x}_n \\ | & | & | & | \end{bmatrix} \quad (1)$$

The CNN with max pooling architecture (Collobert et al., 2011; Kim, 2014) is then used for modelling the sentence. Specifically, a convolution operation using a filter $\mathbf{w} \in \mathbb{R}^{d \times h}$ is applied to a window of h words to extract a feature c_i from a window of words $\mathbf{x}_{i:i+h-1}$ as follows:

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

where f is an activation function, such as tanh or rectifier linear unit (ReLU) (Nair and Hinton, 2010), and $b \in \mathbb{R}$ is a bias.

The filter $\mathbf{w}$ is convolved over the sequence of words represented in the sentence matrix $\mathbf{S}$ to create a feature matrix $\mathbf{C}$. In order to capture the most important features, max pooling is applied to take the maximum value of each row in the matrix $\mathbf{C}$:

$$\mathbf{c}_{max} = \begin{bmatrix} max(\mathbf{C}_{1,:}) \\ \vdots \\ max(\mathbf{C}_{d,:}) \end{bmatrix} \quad (3)$$

This fixed sized vector $\mathbf{c}_{max}$ forms a fully connected layer, before passing to a softmax function for classification. Note that multiple filters (e.g. using different window sizes) can be used to extract features for the fully connected layer.

3 Modelling the Combination of Word Embeddings

We investigate two approaches to model the combination of generic and target domain word embeddings in the described CNN architecture.

3.1 Vector Concatenation

The first approach (namely, *vector concatenation*) is to concatenate vectors from the two embeddings when generating the sentence matrix $\mathbf{S}$ (i.e. at the input layer). In particular, each word vector $\mathbf{x}_i$ in the sentence matrix $\mathbf{S}$ becomes the concatenation

[3] https://semanticnetwork.nlm.nih.gov/

of the vectors from both generic and target domain embeddings corresponding to that word. This allows the filter $\mathbf{w}$ to learn the importance of each dimension of both embeddings[4].

3.2 Combining when Forming the Fully Connected Layer

The second approach (namely, *fully connected layer combination*) models the combination of the word embeddings when forming the fully connected layer before applying softmax for classification. Indeed, we apply the convolution operation (i.e. the convolutional layer in Figure 1) on two different sentence matrices, each of which is created using either the generic or the target domain embeddings. Then, the extracted features are concatenated at a single fully connected layer before applying softmax. This enables the model to learn the importance of each feature from both embeddings directly, before allowing the softmax to take into account the extracted features. Intuitively, this approach should be more effective than the first approach, as it allows more parameters to be learned directly based on the effectiveness of the word vectors from each of the embeddings.

4 Experimental Setup

4.1 Test Collection

To evaluate our approaches, we use two different test collections, which represent domain-specific tasks where existing target domain documents for training word embeddings may be limited. First, the adverse drug reaction (ADR) tweet collection (Ginn et al., 2014) contains 5,250 Twitter messages[5] that can be classified as ADR and non-ADR discussions. Second, the movie review collection (Pang and Lee, 2005)[6] consists of 10,662 sentences that can be classified as having a positive or a negative meaning. On average, a sentence contains 20 terms. For both collections, we report the performance based on the accuracy measure (Pang and Lee, 2005; Ginn et al., 2014), and use paired t-test ($p < 0.05$) to measure the significant difference between the performance achieved by the proposed approaches and the baselines.

[4]The size of the filter $\mathbf{w} \in \mathbb{R}^{d_* \times h}$ depends on the dimension d_* of the concatenated vectors.

[5]We have a smaller dataset than the original paper because some tweets can no longer be accessed via Twitter API.

[6]https://www.cs.cornell.edu/people/pabo/movie-review-data

4.2 Pre-trained Word Embeddings

As a representative of generic word embeddings, we use the publicly available 300-dimension embeddings (vocabulary size of 3M) that were induced from 100 billion words from Google News using word2vec[7], which has been shown to be effective for several tasks (Baroni et al., 2014; Kim, 2014). For target domain embeddings, we use the skip-gram model from word2vec (using default parameters) to create 300-dimension word embeddings from two different publicly available corpora, which are considerably smaller than the Google News. Specifically, the first corpus, representing the target domain corpus of the ADR tweet classification task, contains 854M words from 119k medical articles from BioMed Central. The vocabulary size is 1.3M. For the movie review classification task, we use 24M words of 28k movie reviews from the IMDb archive[8] for inducing the target domain embedding (vocabulary size of 63k). In addition, we use a vector of random values sampled from $[-0.25, 0.25]$ to represent a word that does not exist in any embedding.

4.3 Hyper-parameters and Training Regime

We set the hyper-parameters of CNN in our approaches and the baselines following Kim (2014), whose system achieved state-of-the-art performances on several sentence classification tasks, including the movie review classification task evaluated in this paper. Indeed, we use ReLU as activation functions, and use the filter $\mathbf{w}$ with the window size (h) of 3, 4 and 5, each of which with 100 feature maps. We also apply dropout (dropout rate 0.5) (Srivastava et al., 2014) and L_2 regularisation of the weight vectors at the fully connected layer.

We conduct experiments using 10-fold cross validation. The CNN model is trained over a mini-batch of size 50 by back-propagation. The stochastic gradient decent is performed using Adadelta update rule (Zeiler, 2012) to minimise the negative log-likelihood of correct predictions.

5 Experimental Results

We compare the performance of our approaches, i.e. vector concatenation (Section 3.1) and fully connected layer combination (Section 3.2), with that of the effective CNN model of Kim (2014) (denoted, *simple CNN*). Note that we use the *static*

[7]https://code.google.com/p/word2vec/

[8]Downloaded from http://www.cs.cornell.edu/people/pabo/movie-review-data/polarity_html.zip.

Approach	Word Embeddings	Accuracy	
		ADR Tweets	Movie Review
Simple CNN (Kim, 2014)	Random	87.97	72.41
Simple CNN (Kim, 2014)	Generic	88.47	80.56*
Simple CNN (Kim, 2014)	Domain	88.75*	80.88*
Vector Concatenation	Generic+Domain	88.85*	81.29*
Vector Concatenation	Random+Random	87.96	72.28
Vector Concatenation	Generic+Generic	88.76	80.69
Vector Concatenation	Domain+Domain	88.88	80.61
Fully Connected Layer Combination	Generic+Domain	**89.74**$^{*\circ\bullet}$	**81.59**$^{*\circ\bullet}$
Fully Connected Layer Combination	Random+Random	88.61	72.57
Fully Connected Layer Combination	Generic+Generic	89.47	80.54
Fully Connected Layer Combination	Domain+Domain	89.21	80.81

Table 1: The accuracy performance of the proposed approaches and the *simple CNN* baselines (Kim, 2014). Significant differences ($p < 0.05$, paired t-test) compared to the simple CNN baselines with the *Random*, *Generic* and *Domain* word embeddings, are denoted *, $^{\circ}$ and $^{\bullet}$, respectively.

variant of the CNN model, which does not allow the input embeddings to be updated during training, as we aim to investigate the performance when using original embeddings[9]. In addition to the pre-trained embeddings described in Section 4.2, we use 300-dimension randomly generated word embeddings, as an alternative baseline.

Table 1 reports the accuracy performance of our approaches and the simple CNN baselines on the ADR tweet and movie review classification tasks. We first compare the effectiveness of the simple CNN baselines when applied with different word embeddings. For both tasks, the simple CNN with the target domain word embeddings (accuracy 88.75% and 80.88%) outperforms the simple CNN with either the generic (accuracy 88.47% and 80.56%) or the random (accuracy 87.97% and 72.41%) word embeddings. The performance differences between using the target domain and the random word embeddings are statistically significant ($p < 0.05$) for both tasks. These results show the importance of target domain embedding for the simple CNN on the classification tasks.

Next, we discuss the performance of our two proposed approaches. As shown in Table 1, *Fully Connected Layer Combination (Generic+Domain)* performs better than all of the other approached reported in this paper for both the ADR tweet (accuracy 89.74%) and movie review (accuracy 81.59%) classification tasks. Importantly, it significantly ($p < 0.05$) outperforms the simple CNN baselines that use either the random, generic or target domain word embeddings for both tasks. Meanwhile, *Vector Concatenation (Generic+Domain)* also outperforms all of the simple CNN baselines. These support our hypothesis that exploiting both the generic and target domain word embeddings further improves the performance of CNN for sentence classification.

To further support that our approaches are effective because of exploiting both generic and target domain embeddings rather than because of allowing the model to learn more parameters, we compare our approaches with another set of baselines that use either the generic, target domain, or random embedding twice in both of our proposed approaches. We observe that *Fully Connected Layer Combination (Generic+Domain)* outperforms all of its corresponding baselines, e.g. *Domain+Domain*, for both tasks. The same trends of performance are also observed for the vector concatenation approach, excepting that *Vector Concatenation (Domain+Domain)* marginally outperforms *Vector Concatenation (Generic+Domain)* on the ADR tweet classification task.

6 Conclusions

We have shown the potential of incorporating generic and target domain embeddings in CNN for sentence classification. This provides an alternative method for exploiting generic word embeddings for a given task, where existing domain knowledge or corpora for creating word embeddings are limited, as well as avoiding inducing new word embeddings from a large number of target domain documents for individual tasks. We proposed two approaches that modelled the combination of the two embeddings at the input layer and the fully connected layer of a CNN model. Our experimental results conducted on the ADR tweet and movie review classification tasks showed that both approaches significantly improved the performance over a strong CNN baseline.

[9]The performances of both Kim's and our approaches will further improve, if we allow the embeddings to be updated.

Acknowledgements

The authors acknowledge the support of the EPSRC (grant number EP/M005089/1).

References

Marco Baroni, Georgiana Dinu, and Germán Kruszewski. 2014. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *ACL*, pages 238–247.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *ICML*, pages 160–167.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12:2493–2537.

Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard H. Hovy, and Noah A. Smith. 2015. Retrofitting word vectors to semantic lexicons. In *NAACL*, pages 1606–1615.

Rachel Ginn, Pranoti Pimpalkhute, Azadeh Nikfarjam, Apurv Patki, Karen OConnor, Abeed Sarker, Karen Smith, and Graciela Gonzalez. 2014. Mining twitter for adverse drug reaction mentions: a corpus and classification benchmark. In *The fourth workshop on building and evaluating resources for health and biomedical text processing*.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *KDD*, pages 168–177.

Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. In *NIPS*, pages 2042–2050.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. In *ACL*, pages 655–665.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *EMNLP*, pages 1746–1751.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional Random Fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, pages 282–289.

Omer Levy and Yoav Goldberg. 2014. Dependency-based word embeddings. In *ACL*, pages 302–308.

Nut Limsopatham and Nigel Collier. 2015a. Adapting phrase-based machine translation to normalise medical terms in social media messages. In *EMNLP*, pages 1675–1680.

Nut Limsopatham and Nigel Collier. 2015b. Towards the semantic interpretation of personal health messages from social media. In *Proceedings of the ACM First International Workshop on Understanding the City with Urban Informatics*, UCUI '15, pages 27–30.

Nut Limsopatham and Nigel Collier. 2016. Normalising medical concepts in social media texts by learning semantic representation. In *ACL*.

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

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

TH Muneeb, Sunil Kumar Sahu, and Ashish Anand. 2015. Evaluating distributed word representations for capturing semantics of biomedical concepts. In *BioNLP*, pages 158–163.

Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *ICML*, pages 807–814.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *ACL*, pages 115–124.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. GloVe: Global vectors for word representation. In *EMNLP*, pages 1532–1543.

Isabel Segura-Bedmar, Vıctor Suárez-Paniagua, and Paloma Martınez. 2015. Exploring word embedding for drug name recognition. In *LOUHI*, pages 64–72.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *ACL*, pages 384–394.

Chang Xu, Yalong Bai, Jiang Bian, Bin Gao, Gang Wang, Xiaoguang Liu, and Tie-Yan Liu. 2014. Rc-net: A general framework for incorporating knowledge into word representations. In *CIKM*, pages 1219–1228.

Mo Yu and Mark Dredze. 2014. Improving lexical embeddings with semantic knowledge. In *ACL*, pages 545–550.

Matthew D Zeiler. 2012. ADADELTA: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*.