Task-oriented Domain-specific Meta-Embedding for Text Classification

Xin Wu[1,2], **Yi Cai**[1,2]*, **Qing Li**[3], **Tao Wang**[4], **Kai Yang**[5]
[1]School of Software Engineering, South China University of Technology, Guangzhou, China
[2]Key Laboratory of Big Data and Intelligent Robot (South China University of Technology), Ministry of Education
[3]Department of Computing, The Hong Kong Polytechnic University , Hong Kong, China
[4]Department of Biostatistics & Health Informatics, King's College, London, UK
[5]City University of Hong Kong, Hong Kong, China
ycai@scut.edu.cn

Abstract

Meta-embedding learning, which combines complementary information in different word embeddings, have shown superior performances across different Natural Language Processing tasks. However, domain-specific knowledge is still ignored by existing meta-embedding methods, which results in unstable performances across specific domains. Moreover, the importance of general and domain word embeddings is related to downstream tasks, how to regularize meta-embedding to adapt downstream tasks is an unsolved problem. In this paper, we propose a method to incorporate both domain-specific and task-oriented information into meta-embeddings. We conducted extensive experiments on four text classification datasets and the results show the effectiveness of our proposed method.

1 Introduction

Building semantic representations (Zhao et al., 2017; Li et al., 2019; Neill and Bollegala, 2020) of words is a vital procedure in various Natural Language Processing (NLP) tasks. Over recent years, many pre-trained word embeddings have emerged, such as pre-trained Word2Vec (Mikolov et al., 2013) and pre-trained Glove (Pennington et al., 2014). Despite their usefulness, some previous works find that the performance of different pre-trained word embeddings has significant variation for different tasks (Chen et al., 2013; Hill et al., 2014). To obtain a stable and better performance, Yin and Schütze (2015) proposed the ***meta-embedding*** learning task that aims to obtain a robust and superior word embedding (*i.e.*, meta-embedding) by combining the different pre-trained word embeddings.

Most previous meta-embedding methods neglect the importance of domain-specific information and use the same embedding for each word in all domain-specific datasets (Bollegala and Bao, 2018; Coates and Bollegala, 2018; Bollegala et al., 2017). It is beneficial to incorporate domain-specific information into general word embeddings and provide different word representations for different domains, which has been shown to improve the performance in some other tasks (Bollegala et al., 2015; Xu et al., 2018).

This leads us to explore how to combine general and domain-specific information in meta-embedding learning. Intuitively, the importance of the general and domain embeddings depends on a specific domain. For example, in the computer domain, for the domain-specific words (*e.g.*, "mouse"), we should preserve their domain information but discard their general information. On the other hand, some general words (*e.g.*, "we", "people") may not be able to get a high-quality domain embedding due to the insufficient domain data, in this situation, their general word embeddings are preferable. However, most previous meta-embedding methods are unsupervised, it is hard to learn which embedding is preferable. We consider that it is necessary to use the supervision from a downstream task to address this limitation. Specifically, we focus on text classification (TC) and use the words' category distributions of a TC dataset to guide the meta-embedding learning process.

In this paper, we propose a supervised autoencoder method, named **Task-oriented Domain-specific AutoEncoded Meta-Embedding (TDAEME)**, to learn meta-embedding for text classification. TDAEME combines both general and domain word embeddings in a supervised manner, which is implemented by a supervised autoencoder. Specifically, TDAEME predicts the words' category distribution. This makes the downstream classifier easier to extract useful information from our task-oriented domain-specific

*Corresponding author

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 3508–3513,
November 16–20, 2020.

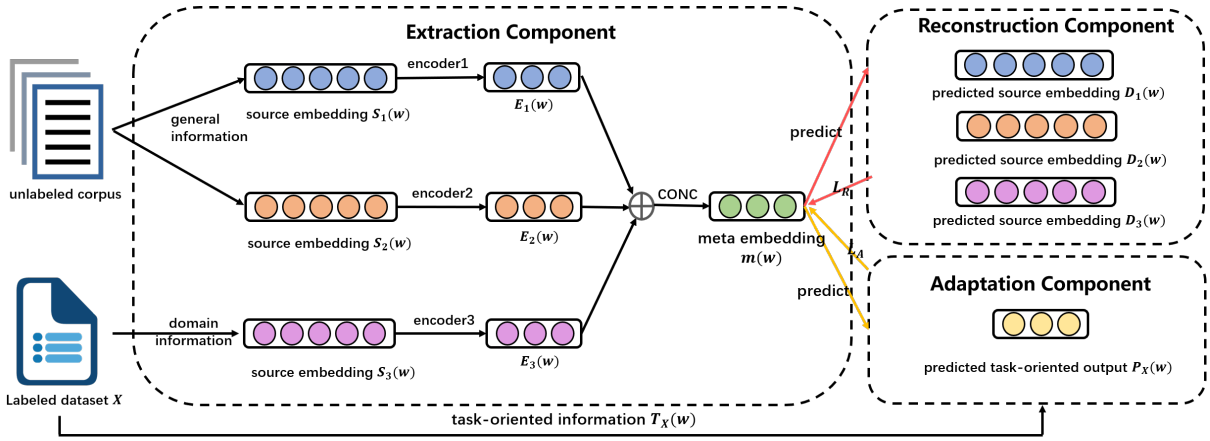

Figure 1: The architecture of Task-oriented Domain-specific AutoEncoded Meta-Embedding (TDAEME).

meta-embedding. We evaluate TDAEME on four text classification datasets, the results demonstrate the effectiveness of our method.

2 Related Work

Yin and Schütze (2015) first proposed a meta-embedding learning method (1TON) to combine the complementary information of multiple pre-trained word embeddings into one meta-embedding. Bollegala and Bao (2018) further improved 1TON by applying an autoencoder framework and three different objective functions to model multiple pre-trained word embeddings. The three new models are called DAEME, CAEME, and AAEME respectively. Bollegala et al. (2017) proposed an unsupervised locally linear method for learning meta-embeddings from a set of source embeddings. However, all the above methods only model the information in pre-trained word embeddings which were trained on unlabeled text but ignore the domain information. One similar work called dynamic meta-embedding which proposed by Kiela et al. (2018) aims to address the meta-embedding learning as a supervised learning paradigm. However, their method is built-in downstream models, which is quite different from our proposed method. Our method is model-independent, the obtained meta-embeddings can be used in any downstream models as features.

One contemporary work also uses the supervised autoencoder method for meta-embedding learning (O'Neill and Bollegala, 2020). However, their motivation and contribution are different from ours. O'Neill and Bollegala (2020) aim to enhance the meta-embedding with words' similarity information, so they use the similarity score between words as the supervision signal in meta-embedding learning, while we focus on a more specifically task (i.e., text classification), our model uses words' categories information as the supervision signal, which is specifically designed for the classification task. In text classification task, words within the same category should be close to each other in the representation space, using similarity information may make two words in different categories get closer (e.g., "learning" and "education" with high similarity but mainly appear in two different categories "AI" and "sociology" respectively).

3 Method

Suppose that we have a word embedding set $S = \{S_1, S_2, ..., S_n\}$ with a vocabulary V; a labeled text classification dataset X which contains a training set X_{train} and a test set X_{test}, we denote its vocabulary as V_X and its categories as C_X with $|C_X| = L$. We aim to learn the word task-oriented domain-specific meta-embedding $m(w)$ for each word $w \in V \cap V_X$. The architecture of TDAEME is visualized in Figure 1

3.1 Extraction Component

The extraction component is used to project different word embeddings into one coherent vector space. For each word w in the source embedding set vocabulary V, $S_i(w)$ denotes the i source embedding of word w, we first use n encoders to extract the semantic information of each source embedding into an d_M dimensional vector space, denote as $E_i(w)$:

$$E_i(w) = f_i(S_i(w)), \quad (1)$$

where f_i is the i encoder function for the i source embedding. Then we compute the task-oriented

domain-specific meta-embedding $m(w)$ of word w:

$$m(w) = E_1(w) \oplus E_2(w) \oplus, ..., \oplus E_n(w), \quad (2)$$

where $\oplus$ is the concatenation operator.

3.2 Reconstruction Component

In this component, we take the $m(w)$ as input, then predict all n source embeddings $D_i(w)$:

$$D_i(w) = g_i(m(w)), \quad (3)$$

where g_i is the i decoder function to predict the i source embedding from the $m(w)$. The objective of this component can be represented as L_R:

$$L_R = \sum_{w \in V} \sum_{i=1}^{n} \lambda_i \| S_i(w) - D_i(w) \|_2^2, \quad (4)$$

where $S_i(w)$ and $D_i(w)$ is the i source and predict embedding of word w, λ_i is a hyperparameter to adapt the weight of different source embeddings.

3.3 Adaption Component

In this component, we make the $m(w)$ predict its category distribution of a downstream dataset. This makes words with the same category would get close in meta-embedding vector space. Formally, For each word w both in vocabulary V (the vocabulary of all source embeddings) and V_X (the vocabulary of the classification dataset X), its category distribution can be defined as $T_X(w)$:

$$T_X(w) = [T_X^{C_1}(w), T_X^{C_2}(w), ..., T_X^{C_L}(w)], \quad (5)$$

$$T_X^{C_j}(w) = \frac{t_X^{C_j}}{\sum_{k=1}^{L} t_X^{C_k}}, \quad (6)$$

where $T_X^{C_j}(w)$ is the document frequency of word w in jth category, $t_X^{C_j}$ is the number of documents that contain w in the class C_j.

An extra decoder is employed to predict the category distribution $P_X(w)$ of word w from the $m(w)$:

$$P_X(w) = [P_X^{C_1}(w), P_X^{C_2}(w), ..., P_X^{L}(w)]. \quad (7)$$

The objective of this component L_A can be represented as:

$$L_A = \sum_{w \in V \cap V_X} \sum_{j=1}^{L} \left\| T_X^{C_j}(w) - P_X^{C_j}(w) \right\|_2^2. \quad (8)$$

3.4 Joint Learning

The extraction component is shared between the reconstruction component and the adaption component, we propose to use the joint learning framework to jointly optimizing L_R and L_A. Then we obtained the final objective function L:

$$L = \alpha L_R + (1 - \alpha) L_A, \quad (9)$$

where α is a hyperparameter to adapt the reconstruction component and the adaption component.

4 Experiments

4.1 Source Word Embeddings

We use the Glove[1] and CBOW[2] as the two **general word embeddings** in our experiments. To obtain the **domain word embeddings**, we use the training set X_{train} of each downstream task to train the corresponding domain word embeddings for each dataset. In this paper, we use cbow model from the Word2Vec open source package2.

Datasets	Type	Train Size	Test Size	Class Num
20News Group	Doc.	16938	1890	20
5Abstracts Group	Doc.	5616	630	5
IMDB	Doc.	45000	5000	2
TREC	Sen.	5452	500	6

Table 1: Statistics of the four datasets.

4.2 Datasets

To evaluate the effectiveness of our proposed model TDAEME, we conduct extensive Experiments on four English text classification datasets: **20NewsGroup**[3] (Lang, 1995), **5AbstractsGroup**[4] (Liu et al., 2018), **IMDB**[5] (Maas et al., 2011), **TREC**[6] (Li and Roth, 2002).The statistics of the datasets are give in the Table 1. We didn't split a validation set, see details in 4.4

4.3 Baseline Methods

We consider the following meta-embedding approaches as baselines: (1) **Concatenation**

[1]http://nlp.stanford.edu/projects/glove/
[2]https://code.google.com/archive/p/word2vec
[3]http://qwone.com/%7ejason/20Newsgroups/
[4]https://github.com/qianliu0708/5AbstractsGroup
[5]https://ai.stanford.edu/%7eamaas/data/sentiment/
[6]http://cogcomp.cs.illinois.edu/Data/QA/QC/

Methods		20NewsGroup	5AbstractsGroup	IMDB	TREC
source embeddings	CBOW	0.695	0.725	0.832	0.754
	Glove	0.665	0.752	0.829	0.772
	domain embeddings	0.43	0.736	0.834	0.608
baseline	CONC	0.759	0.819	0.853	0.822
	AVG	0.730	0.814	0.836	0.784
	DAEME	0.758	**0.857**	0.849	0.852
	CAEME	0.765	0.837	0.85	0.84
	AAEME	0.723	0.851	0.844	0.812
	LLE	0.718	0.795	0.851	0.854
ablation	TDAEME (w/o CBOW)	0.771	0.833	0.861	0.844
	TDAEME (w/o Glove)	0.734	0.819	0.859	0.846
	TDAEME (w/o domain)	0.770	0.825	0.859	0.856
	TDAEME (w/o adaption)	0.763	0.848	0.862	0.866
	TDAEME	**0.788****	**0.857**	**0.865***	**0.894****

Table 2: Experimental results. Bold scores are the best overall. w/o represents removing one component or one source embedding, while remaining other components. *, ** indicates p-value<0.05, <0,01, respectively.

(CONC) Yin and Schütze (2015) propose that the concatenation of the source embeddings is an effective method for creating meta-embeddings. (2) **Averaging (AVG)** Coates and Bollegala (2018) proposed averaging the source word embeddings for a word as a method for creating meta-embeddings without increasing the representation dimensionality. (3) **origin AEMEs** Bollegala and Bao (2018) proposed three autoencoder-based approaches DAEME, CAEME, and AAEME for learning meta-embeddings from multiple pre-trained source embeddings.We use the code[7] released by the authors in our experiments. (4) **LLE** Bollegala et al. (2017) proposed an unsupervised locally linear method for learning meta-embeddings from a set of source embeddings. We use the code[8] released by the authors in our experiments.

4.4 Experimental Settings

We use the average of word embeddings to represent the document. We trained a linear classifier using Liblinear (Fan et al., 2008) to test the classification performance of each embedding. Since the goal is to evaluate the embeddings, so we didn't tune the hyperparameters of the classifier on a validation set and just evaluate the test set performance with default hyperparameters. To train our proposed model TDAEME, we use a linear neural layer with the ReLU (Nair and Hinton, 2010) activation function as an encoder and a linear neural layer as a decoder. We employ Adam (Kingma and Ba, 2014) with mini-batches of size 128 and 0.001 learning rate as an optimizer. We also applied masking noises (Vincent et al., 2010) to randomly set 0.05% of the input elements to zero. α is set to 1e-4. We manually tuned the hyperparameters of TDAEME according to the training loss (*i.e.*, equation 4 8 9) of TDAEME. The computing infrastructure we used is a PC with GTX 980Ti.

4.5 Result

Overall Performance We use accuracy [9] as metric in our experiments. Table 2 shows the evaluation results. Compared with two general source embeddings, meta-embedding learning methods perform better in most cases, which demonstrates the effectiveness of meta-embedding methods. Moreover, fine-tuning meta-embedding learning methods (*i.e.*, AEMEs) have better performance than none-learning methods (*i.e.*, CONC and AVG). Compared with three origin AEME models and LLE, our proposed method TDAEME can make a further improvement in the text classification task, which demonstrates the effectiveness of the domain-specifc and task-orietend information.

Ablation The last 5 rows in Table 2 shows the ablation results. In most cases, combining one more high-quality general word embeddings will never harm the performance. While the results of the last two ablation methods indicate that both

[7] https://github.com/CongBao/AutoencodedMetaEmbedding
[8] https://github.com/LivNLP/LLE-MetaEmbed
[9] we use the code from scikit-learn.org

the domain embeddings and the adaption component provide a significant boost compared to the raw AEMEs. Moreover, TDAEME achieves the best results among all ablation methods. This indicates the domain-specific and task-oriented information are beneficial to each other, our joint learning method can successfully model these two types of information.

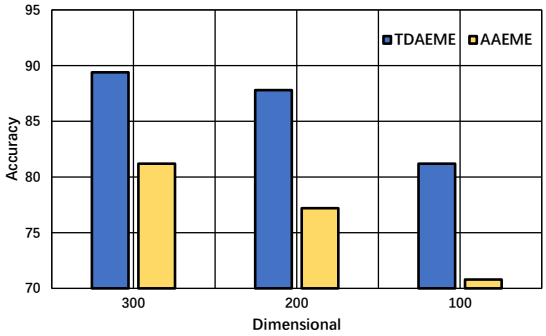

Figure 2: Experiment results of different meta-embedding dimensions on TREC dataset. The axes represent accuracy and dimension respectively.

Impact of Dimensional We also conducted an experiment on meta-embedding dimensionalities. We investigate the performance of AAEME and TDAEME on TREC dataset with 100, 200, and 300 meta-embedding dimensions respectively. The results are shown in Figure 2. We find that TDAEME outperforms AAEME in all cases and TDAEME is less sensitive to dimension reduction than AAEME.

	TDAEME	ELMo+SVM
IMDB	0.865	0.86
5AbastractsGroup	0.857	0.857
20NewsGroup	0.788	0.786

Table 3: Experiments results of TDAEME and ELMo+SVM.

Compared with Contextualized Embeddings Contextualized Embeddings such as BERT, ELMo can outperform previous state-of-the-art models on multiple natural language understanding (NLU) benchmarks. We conduct an experiment to compared our TDAEME with ELMo (Peters et al., 2018). To make a fair comparsion, we use ELMo to get sentence embedding, and performance classification with the same SVM classifier. Table 3 shows the results. We observe that our TDAEME can achieve competitive performance against the contextualized embeddings.

5 Conclusion

In this paper, we propose a meta-embedding learning approach called Task-oriented Domain-specific Autoencoded Meta-Embedding (TDAEME), which leverages task-oriented supervision to improve the combination of general and domain embeddings. We conducted experiments on four text classification datasets and the results show the effectiveness of our proposed method.

Acknowledgement

This work was supported by the Fundamental Research Funds for the Central Universities, SCUT (No.2017ZD048, D2182480), the Science and Technology Planning Project of Guangdong Province (No.2017B050506004), the Science and Technology Programs of Guangzhou (No.201704030076, 201802010027, 201902010046), National Natural Science Foundation of China (62076100) and the Hong Kong Research Grants Council (project no. C1031-18G).

References

Danushka Bollegala and Cong Bao. 2018. Learning word meta-embeddings by autoencoding. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1650–1661.

Danushka Bollegala, Kohei Hayashi, and Ken-ichi Kawarabayashi. 2017. Think globally, embed locally—locally linear meta-embedding of words. *arXiv preprint arXiv:1709.06671*.

Danushka Bollegala, Takanori Maehara, and Ken-ichi Kawarabayashi. 2015. Unsupervised cross-domain word representation learning. *arXiv preprint arXiv:1505.07184*.

Yanqing Chen, Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2013. The expressive power of word embeddings. *arXiv preprint arXiv:1301.3226*.

Joshua Coates and Danushka Bollegala. 2018. Frustratingly easy meta-embedding – computing meta-embeddings by averaging source word embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 194–198, New Orleans, Louisiana. Association for Computational Linguistics.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. Liblinear: A

library for large linear classification. *Journal of machine learning research*, 9(Aug):1871–1874.

Felix Hill, KyungHyun Cho, Sebastien Jean, Coline Devin, and Yoshua Bengio. 2014. Not all neural embeddings are born equal. *arXiv preprint arXiv:1410.0718*.

Douwe Kiela, Changhan Wang, and Kyunghyun Cho. 2018. Dynamic meta-embeddings for improved sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1466–1477, Brussels, Belgium. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Ken Lang. 1995. Newsweeder: Learning to filter netnews. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 331–339.

Bofang Li, Aleksandr Drozd, Yuhe Guo, Tao Liu, Satoshi Matsuoka, and Xiaoyong Du. 2019. Scaling word2vec on big corpus. *Data Science and Engineering*, 4(2):157–175.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Qian Liu, Heyan Huang, Yang Gao, Xiaochi Wei, Yuxin Tian, and Luyang Liu. 2018. Task-oriented word embedding for text classification. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2023–2032, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

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.

Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814.

James O' Neill and Danushka Bollegala. 2020. Meta-embedding as auxiliary task regularization.

James O'Neill and Danushka Bollegala. 2020. Meta-embedding as auxiliary task regularization. In *ECAI 2020 - 24th European Conference on Artificial Intelligence, 29 August-8 September 2020, Santiago de Compostela, Spain, August 29 - September 8, 2020 - Including 10th Conference on Prestigious Applications of Artificial Intelligence (PAIS 2020)*, volume 325 of *Frontiers in Artificial Intelligence and Applications*, pages 2124–2131. IOS Press.

Jeffrey Pennington, Richard Socher, and Christopher 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, Doha, Qatar. Association for Computational Linguistics.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of NAACL-HLT*, pages 2227–2237.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(Dec):3371–3408.

Hu Xu, Bing Liu, Lei Shu, and Philip S Yu. 2018. Double embeddings and cnn-based sequence labeling for aspect extraction. *arXiv preprint arXiv:1805.04601*.

Wenpeng Yin and Hinrich Schütze. 2015. Learning meta-embeddings by using ensembles of embedding sets. *arXiv preprint arXiv:1508.04257*.

Zhe Zhao, Tao Liu, Shen Li, Bofang Li, and Xiaoyong Du. 2017. Guiding the training of distributed text representation with supervised weighting scheme for sentiment analysis. *Data Science and Engineering*, 2(2):178–186.