How to Fine-Tune BERT for Text Classification?

Chi Sun, Xipeng Qiu(✉), Yige Xu, and Xuanjing Huang

Shanghai Key Laboratory of Intelligent Information Processing,
School of Computer Science, Fudan University,
825 Zhangheng Road, Shanghai, China
{sunc17,xpqiu,ygxu18,xjhuang}@fudan.edu.cn

Abstract. Language model pre-training has proven to be useful in learning universal language representations. As a state-of-the-art language model pre-training model, BERT (Bidirectional Encoder Representations from Transformers) has achieved amazing results in many language understanding tasks. In this paper, we conduct exhaustive experiments to investigate different fine-tuning methods of BERT on text classification task and provide a general solution for BERT fine-tuning. Finally, the proposed solution obtains new state-of-the-art results on eight widely-studied text classification datasets.

Keywords: Transfer learning · BERT · Text classification

1 Introduction

Text classification is a classic problem in Natural Language Processing (NLP). The task is to assign predefined categories to a given text sequence. An important intermediate step is the text representation. Previous work uses various neural models to learn text representation, including convolution models, recurrent models, and attention mechanisms.

Recently, pre-trained language models have shown to be useful in learning common language representations by utilizing a large amount of unlabeled data: e.g., ELMo [20], OpenAI GPT [22] and BERT [6]. Among them, BERT is based on a multi-layer bidirectional Transformer [24] and is trained on plain text for masked word prediction and next sentence prediction tasks.

Although BERT has achieved amazing results in many natural language understanding (NLU) tasks, its potential has yet to be fully explored. There is little research to enhance BERT to improve the performance on target tasks further.

In this paper, we investigate how to maximize the utilization of BERT for the text classification task. We explore several ways of fine-tuning BERT to enhance its performance on text classification task. We design exhaustive experiments to make a detailed analysis of BERT.

M. Sun et al. (Eds.): CCL 2019, LNAI 11856, pp. 194–206, 2019.
https://doi.org/10.1007/978-3-030-32381-3_16

The contributions of our paper are as follows:

- We propose a general solution to fine-tune the pre-trained BERT model, which includes three steps: (1) further pre-train BERT on within-task training data or in-domain data; (2) optional fine-tuning BERT with multi-task learning if several related tasks are available; (3) fine-tune BERT for the target task.
- We also investigate the fine-tuning methods for BERT on target task, including layer-wise learning rate, catastrophic forgetting, and few-shot learning problems.
- We achieve the new state-of-the-art results on eight widely-studied text classification datasets.

2 Related Work

Borrowing the learned knowledge from the other tasks has a rising interest in the field of NLP. We briefly review two related approaches: language model pre-training and multi-task Learning.

2.1 Language Model Pre-training

Pre-trained word embeddings [18,19], as an important component of modern NLP systems can offer significant improvements over embeddings learned from scratch. The generalization of word embeddings, such as sentence embeddings [8,14] or paragraph embeddings [9], are also used as features in downstream models.

Peters et al. [20] concatenate embeddings derived from language model as additional features for the main task and advance the state-of-the-art for several major NLP benchmarks. In addition to pre-training with unsupervised data, transfer learning with a large amount of supervised data can also achieve good performance, such as natural language inference [4] and machine translation [16].

More recently, the method of pre-training language models on a large network with a large amount of unlabeled data and fine-tuning in downstream tasks has made a breakthrough in several natural language understanding tasks, such as OpenAI GPT [22] and BERT [6]. Dai and Le [5] use language model fine-tuning but overfit with 10k labeled examples while Howard and Ruder [7] propose ULMFiT and achieve state-of-the-art results in the text classification task. BERT is pre-trained on *Masked Language Model Task* and *Next Sentence Prediction Task* via a large cross-domain corpus. Unlike previous bidirectional language models (biLM) limited to a combination of two unidirectional language models (i.e., left-to-right and right-to-left), BERT uses a Masked Language Model to predict words which are randomly masked or replaced. BERT is the first fine-tuning based representation model that achieves state-of-the-art results for a range of NLP tasks, demonstrating the enormous potential of the fine-tuning method. In this paper, we have further explored the BERT fine-tuning method for text classification.

2.2 Multi-task Learning

Multi-task learning [1,3] is another relevant direction. Rei [23] and Liu et al. [11] use this method to train the language model and the main task model jointly. Liu et al. [13] extend the MT-DNN model originally proposed in [12] by incorporating BERT as its shared text encoding layers. MTL requires training tasks from scratch every time, which makes it inefficient and it usually requires careful weighing of task-specific objective functions [2]. However, we can use multi-task BERT fine-tuning to avoid this problem by making full use of the shared pre-trained model.

3 BERT for Text Classification

BERT-base model contains an encoder with 12 Transformer blocks, 12 self-attention heads, and the hidden size of 768. BERT takes an input of a sequence of no more than 512 tokens and outputs the representation of the sequence. The sequence has one or two segments that the first token of the sequence is always `[CLS]` which contains the special classification embedding and another special token `[SEP]` is used for separating segments.

For text classification tasks, BERT takes the final hidden state $\mathbf{h}$ of the first token `[CLS]` as the representation of the whole sequence. A simple softmax classifier is added to the top of BERT to predict the probability of label c:

$$p(c|\mathbf{h}) = \text{softmax}(W\mathbf{h}), \tag{1}$$

where W is the task-specific parameter matrix. We fine-tune all the parameters from BERT as well as W jointly by maximizing the log-probability of the correct label.

4 Methodology

When we adapt BERT to NLP tasks in a target domain, a proper fine-tuning strategy is desired. In this paper, we look for the proper fine-tuning methods in the following three ways.

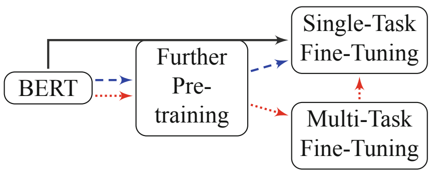


Fig. 1. Three general ways for fine-tuning BERT, shown with different colors. (Color figure online)

(1) **Fine-Tuning Strategies:** When we fine-tune BERT for a target task, there are many ways to utilize BERT. For example, the different layers of BERT capture different levels of semantic and syntactic information, which layer is better for a target task? How we choose a better optimization algorithm and learning rate?
(2) **Further Pre-training:** BERT is trained in the general domain, which has a different data distribution from the target domain. A natural idea is to further pre-train BERT with target domain data.
(3) **Multi-task Fine-Tuning:** Without pre-trained LM models, multi-task learning has shown its effectiveness of exploiting the shared knowledge among the multiple tasks. When there are several available tasks in a target domain, an interesting question is whether it still bring benefits to fine-tune BERT on all the tasks simultaneously.

Our general methodology of fine-tuning BERT is shown in Fig. 1.

4.1 Fine-Tuning Strategies

Different layers of a neural network can capture different levels of syntactic and semantic information [7,27].

To adapt BERT to a target task, we need to consider the overfitting problem. A better optimizer with an appropriate learning rate is desired. Intuitively, the lower layer of the BERT model may contain more general information. We can fine-tune them with different learning rates.

Following [7], we split the parameters θ into $\{\theta^1, \cdots, \theta^L\}$ where θ^l contains the parameters of the l-th layer of BERT. Then the parameters are updated as follows:

$$\theta_t^l = \theta_{t-1}^l - \eta^l \cdot \nabla_{\theta^l} J(\theta), \tag{2}$$

where η^l represents the learning rate of the l-th layer.

We set the base learning rate to η^L and use $\eta^{k-1} = \xi \cdot \eta^k$, where ξ is a decay factor and less than or equal to 1. When $\xi < 1$, the lower layer has a lower learning rate than the higher layer. When $\xi = 1$, all layers have the same learning rate, which is equivalent to the regular stochastic gradient descent (SGD). We will investigate these factors in Sect. 5.3.

4.2 Further Pre-training

The BERT model is pre-trained in the general-domain corpus. For a text classification task in a specific domain, such as movie reviews, its data distribution may be different from BERT. Therefore, we can further pre-train BERT with masked language model and next sentence prediction tasks on the domain-specific data. Three further pre-training approaches are performed:

(1) Within-task pre-training, in which BERT is further pre-trained on the training data of a target task.
(2) In-domain pre-training, in which the pre-training data is obtained from the same domain of a target task. For example, there are several different sentiment classification tasks, which have a similar data distribution. We can further pre-train BERT on the combined training data from these tasks.
(3) Cross-domain pre-training, in which the pre-training data is obtained from both the same and other different domains to a target task.

We will investigate these different approaches to further pre-training in Sect. 5.4.

4.3 Multi-task Fine-Tuning

Multi-task Learning is also an effective approach to share the knowledge obtained from several related supervised tasks. Similar to [13], we also use fine-tune BERT in multi-task learning framework for text classification.

All the tasks share the BERT layers and the embedding layer. The only layer that does not share is the final classification layer, which means that each task has a private classifier layer. The experimental analysis is in Sect. 5.5.

Table 1. Statistics of eight text classification datasets.

Dataset	Classes	Type	Average lengths	Max lengths	Train samples	Test samples
IMDb [15]	2	Sentiment	292	3,045	25,000	25,000
Yelp P. [28]	2	Sentiment	177	2,066	560,000	38,000
Yelp F. [28]	5	Sentiment	179	2,342	650,000	50,000
TREC [25]	6	Question	11	39	5,452	500
Yahoo! Answers [28]	10	Question	131	4,018	1,400,000	60,000
AG's News [28]	4	Topic	44	221	120,000	7,600
DBPedia [28]	14	Topic	67	3,841	560,000	70,000
Sogou News [28]	6	Topic	737	47,988	54,000	6,000

5 Experiments

We investigate the different fine-tuning methods for seven English and one Chinese text classification tasks. We use the base BERT models: the uncased BERT-base model[1] and the Chinese BERT-base model[2] respectively.

[1] https://storage.googleapis.com/bert_models/2018_10_18/uncased_L-12_H-768_A-12.zip.
[2] https://storage.googleapis.com/bert_models/2018_11_03/chinese_L-12_H-768_A-12.zip.

5.1 Datasets

We evaluate our approach on eight widely-studied datasets. These datasets have varying numbers of documents and varying document lengths, covering three common text classification tasks: sentiment analysis, question classification, and topic classification. We show the statistics for each dataset in Table 1.

Data Pre-processing. Following [6], we use WordPiece embeddings [26] with a 30,000 token vocabulary and denote split word pieces with ##. So the statistics of the length of the documents in the datasets are based on the word pieces. For further pre-training with BERT, we use spaCy[3] to perform sentence segmentation in English datasets and we use "。", "?" and "!" as separators when dealing with the Chinese Sogou News dataset.

5.2 Hyperparameters

We use the BERT-base model [6] with a hidden size of 768, 12 Transformer blocks [24] and 12 self-attention heads. We further pre-train with BERT on 1 TITAN Xp GPU, with a batch size of 32, max squence length of 128, learning rate of 5e−5, train steps of 100,000 and warm-up steps of 10,000.

We fine-tune the BERT model on 4 TITAN Xp GPUs and set the batch size to 24 to ensure that the GPU memory is fully utilized. The dropout probability is always kept at 0.1. We use Adam with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. We use *slanted triangular learning rates* [7], the base learning rate is 2e−5, and the warm-up proportion is 0.1. We empirically set the max number of the epoch to 4 and save the best model on the validation set for testing.

5.3 Exp-I: Investigating Different Fine-Tuning Strategies

In this subsection, we use the IMDb dataset to investigate the different fine-tuning strategies. The official pre-trained model is set as the initial encoder[4].

Layer-Wise Decreasing Layer Rate. Table 2 show the performance of different base learning rate and decay factors (see Eq. (2)) on IMDb dataset. We find that assign a lower learning rate to the lower layer is effective to fine-tuning BERT, and an appropriate setting is $\xi = 0.95$ and lr = 2.0e−5.

Catastrophic Forgetting. Catastrophic forgetting [17] is usually a common problem in transfer learning, which means the pre-trained knowledge is erased during learning of new knowledge. Therefore, we also investigate whether BERT suffers from the catastrophic forgetting problem.

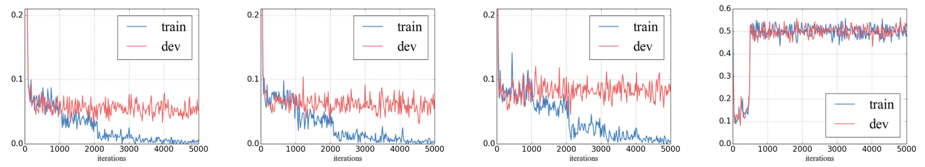
We fine-tune BERT with different learning rates, and the learning curves of error rates on IMDb are shown in Fig. 2.

[3] https://spacy.io/.

[4] https://github.com/google-research/bert.

Table 2. Decreasing layer-wise layer rate.

Learning rate	Decay factor ξ	Test error rates (%)
2.5e−5	1.00	5.52
2.5e−5	0.95	5.46
2.5e−5	0.90	**5.44**
2.0e−5	1.00	5.42
2.0e−5	0.95	**5.40**
2.0e−5	0.90	5.52



(a) lr=2e-5 (b) lr=5e-5 (c) lr=1e-4 (d) lr=4e-4

Fig. 2. Catastrophic forgetting

Table 3. Performance of in-domain and cross-domain further pre-training on seven datasets. Each was further pre-trained for 100k steps. The first column indicates the different further pre-training dataset. "all sentiment" means the dataset consists of all the training datasets in sentiment domain. "all" means the dataset consists of all the seven training datasets.

Domain	Sentiment			Question		Topic	
Dataset	IMDb	Yelp P.	Yelp F.	TREC	Yah. A.	AG's News	DBPedia
IMDb	**4.37**	2.18	29.60	2.60	22.39	5.24	0.68
Yelp P.	5.24	1.92	29.37	2.00	22.38	5.14	**0.65**
Yelp F.	5.18	1.94	29.42	2.40	22.33	5.43	**0.65**
All sentiment	4.88	**1.87**	29.25	3.00	22.35	5.34	0.67
TREC	5.65	2.09	29.35	3.20	22.17	5.12	0.66
Yah. A.	5.52	2.08	29.31	**1.80**	22.38	5.16	0.67
All question	5.68	2.14	29.52	2.20	**21.86**	5.21	0.68
AG's News	5.97	2.15	29.38	2.00	22.32	**4.80**	0.68
DBPedia	5.80	2.13	29.47	2.60	22.30	5.13	0.68
All topic	5.85	2.20	29.68	2.60	22.28	4.88	**0.65**
All	5.18	1.97	**29.20**	2.80	21.94	5.08	0.67
W/o pretrain	5.40	2.28	30.06	2.80	22.42	5.25	0.71

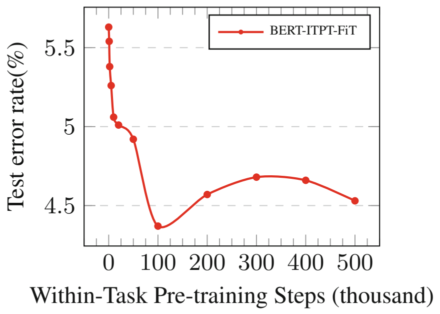


Fig. 3. Benefit of different further pre-training steps on IMDb datasets. BERT-ITPT-FiT means "BERT + withIn-Task Pre-Training + Fine-Tuning".

We find that a lower learning rate, such as 2e−5, is necessary to make BERT overcome the catastrophic forgetting problem. With an aggressive learn rate of 4e−4, the training set fails to converge.

5.4 Exp-II: Investigating the Further Pre-training

Besides, fine-tune BERT with supervised learning, we can further pre-train BERT on the training data by unsupervised masked language model and next sentence prediction tasks. In this section, we investigate the effectiveness of further pre-training. In the following experiments, we use the best strategies in Exp-I during the fine-tuning phase.

Within-Task Further Pre-training. Therefore, we first investigate the effectiveness of within-task further pre-training. We take further pre-trained models with different steps and then fine-tune them with text classification task.

As shown in Fig. 3, the further pre-training is useful to improve the performance of BERT for a target task, which achieves the best performance after 100 K training steps.

In-domain and Cross-Domain Further Pre-training. Besides the training data of a target task, we can further pre-train BERT on the data from the same domain. In this subsection, we investigate whether further pre-training BERT with in-domain and cross-domain data can continue to improve the performance of BERT.

We partition the seven English datasets into three domains: topic, sentiment, and question. The partition way is not strictly correct. Therefore we also conduct extensive experiments for cross-task pre-training, in which each task is regarded as a different domain.

The results is shown in Table 3. We find that almost all further pre-training models perform better on all seven datasets than the original BERT-base model (row 'w/o pretrain' in Table 3). Generally, in-domain pretraining can bring better performance than within-task pretraining. On the small sentence-level TREC dataset, within-task pre-training do harm to the performance while in-domain pre-training which utilizes Yah. A. corpus can achieve better results on TREC.

Comparisons to Previous Models. We compare our model with the feature-based transfer learning methods such as rigion embedding [21] and CoVe [16] and the language model fine-tuning method (ULMFiT) [7], which is the current state-of-the-art for text classification.

We implement BERT-Feat through using the feature from BERT model as the input embedding of the biLSTM with self-attention [10]. The result of BERT-IDPT-FiT corresponds to the row of 'all sentiment', 'all question', and 'all topic' in Table 3, and the result of BERT-CDPT-FiT corresponds to the row of 'all' in it. As is shown in Table 4, BERT-Feat performs better than all other baselines except for ULMFiT. In addition to being slightly worse than BERT-Feat on DBpedia dataset, BERT-FiT outperforms BERT-Feat on the other seven datasets. Moreover, all of the three further pre-training models are better than BERT-FiT model. Using BERT-Feat as a reference, we calculate the average percentage increase of other BERT-FiT models on each dataset. BERT-IDPT-FiT performs best, with an average error rate reduce by 18.57%.

Table 4. Test error rates (%) on eight text classification datasets. BERT-Feat means "BERT as features". BERT-FiT means "BERT + Fine-Tuning". BERT-ITPT-FiT means "BERT + withIn-Task Pre-Training + Fine-Tuning". BERT-IDPT-FiT means "BERT + In-Domain Pre-Training + Fine-Tuning". BERT-CDPT-FiT means "BERT + Cross-Domain Pre-Training + Fine-Tuning".

Model	IMDb	Yelp P.	Yelp F.	TREC	Yah. A.	AG	DBP	Sogou	Avg. Δ
Region Emb. [21]	/	3.60	35.10	/	26.30	7.20	1.10	2.40	/
CoVe [16]	8.20	/	/	4.20	/	/	/	/	/
ULMFiT [7]	4.60	2.16	29.98	3.60	/	5.01	0.80	/	/
BERT-Feat	6.79	2.39	30.47	4.20	22.72	5.92	0.70	2.50	-
BERT-FiT	5.40	2.28	30.06	2.80	22.42	5.25	0.71	2.43	9.22%
BERT-ITPT-FiT	**4.37**	1.92	29.42	3.20	22.38	**4.80**	0.68	**1.93**	16.07%
BERT-IDPT-FiT	4.88	**1.87**	29.25	**2.20**	**21.86**	4.88	**0.65**	/	**18.57%**
BERT-CDPT-FiT	5.18	1.97	**29.20**	2.80	21.94	5.08	0.67	/	14.38%

5.5 Exp-III: Multi-task Fine-Tuning

When there are several datasets for the text classification task, to take full advantage of these available data, we further consider a fine-tuning step with multi-task learning. We use four English text classification datasets (IMDb, Yelp P., AG, and DBP). The dataset Yelp F. is excluded since there is overlap between the test set of Yelp F. and the training set of Yelp P., and two datasets of question domain are also excluded.

Table 5. Test error rates (%) with multi-task fine-tuning.

Method	IMDb	Yelp P.	AG	DBP
BERT-FiT	5.40	2.28	5.25	0.71
BERT-MFiT-FiT	5.36	2.19	5.20	0.68
BERT-CDPT-FiT	5.18	**1.97**	**5.08**	**0.67**
BERT-CDPT-MFiT-FiT	**4.96**	2.06	5.13	**0.67**

Table 5 shows that for multi-task fine-tuning based on BERT, the effect is improved. However, multi-task fine-tuning does not seem to be helpful to BERT-CDPT in Yelp P. and AG. Multi-task fine-tuning and cross-domain pre-training may be alternative methods since the BERT-CDPT model already contains rich domain-specific information, and multi-task learning may not be necessary to improve generalization on related text classification sub-tasks.

5.6 Exp-IV: Few-Shot Text Classification

One of the benefits of the pre-trained model is being able to train a model for downstream tasks within small training data. We evaluate BERT-FiT and BERT-ITPT-FiT on different numbers of training examples. We select a subset of IMDb training data and feed them into BERT-FiT and BERT-ITPT-FiT. We show the result in Fig. 4.

This experiment result demonstrates that further pre-training brings a significant improvement for few-shot text classification. On IMDb dataset, when there are only 100 labeled data for each class, the accuracy of BERT-ITPT-FiT can reach 92%.

5.7 Exp-V: Further Pre-training on BERT Large

In this subsection, we investigate whether the $\text{BERT}_{\text{LARGE}}$ model has similar findings to $\text{BERT}_{\text{BASE}}$. We further pre-train Google's pre-trained $\text{BERT}_{\text{LARGE}}$ model[5] on 1 Tesla-V100-PCIE 32G GPU with a batch size of 24, the max sequence length of 128 and 120 K training steps. For target task classifier BERT fine-tuning, we set the batch size to 24 and fine-tune $\text{BERT}_{\text{LARGE}}$ on 4 Tesla-V100-PCIE 32G GPUs with the max sequence length of 512.

As shown in Table 6, ULMFiT performs better on almost all of the tasks compared to $\text{BERT}_{\text{BASE}}$ but not $\text{BERT}_{\text{LARGE}}$. This changes however with the task-specific further pre-training where even $\text{BERT}_{\text{BASE}}$ outperforms ULMFiT on all tasks. $\text{BERT}_{\text{LARGE}}$ fine-tuning with task-specific further pre-training achieves state-of-the-art results.

[5] https://storage.googleapis.com/bert_models/2018_10_18/uncased_L-24_H-1024_A-16.zip.

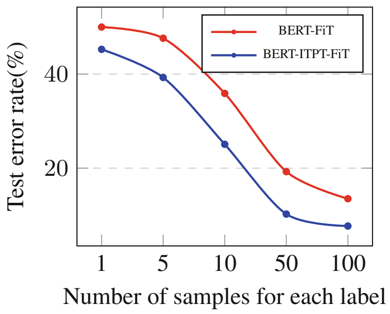


Fig. 4. Few-shot text classification on IMDb dataset

Table 6. Test error rates (%) on five text classification datasets.

Model	IMDb	Yelp P.	Yelp F.	AG	DBP
ULMFiT	4.60	2.16	29.98	5.01	0.80
BERT$_{\text{BASE}}$	5.40	2.28	30.06	5.25	0.71
+ ITPT	4.37	1.92	29.42	4.80	0.68
BERT$_{\text{LARGE}}$	4.86	2.04	29.25	4.86	0.62
+ ITPT	**4.21**	**1.81**	**28.62**	**4.66**	**0.61**

6 Conclusion

In this paper, we conduct extensive experiments to investigate the different approaches to fine-tuning BERT for the text classification task. There are some experimental findings: (1) With an appropriate layer-wise decreasing learning rate, BERT can overcome the catastrophic forgetting problem; (2) Within-task and in-domain further pre-training can significantly boost its performance; (3) A preceding multi-task fine-tuning is also helpful to the single-task fine-tuning, but its benefit is smaller than further pre-training; (4) BERT with further pre-training performs well in few-shot text classification.

With the above findings, we achieve state-of-the-art performances on eight widely studied text classification datasets. In the future, we will probe more insight of BERT on how it works.

Acknowledgments. The research work is supported by China National Key R&D Program No. 2018YFC0831103.

References

1. Caruana, R.: Multitask learning: a knowledge-based source of inductive bias. In: Proceedings of the Tenth International Conference on Machine Learning (1993)
2. Chen, Z., Badrinarayanan, V., Lee, C.Y., Rabinovich, A.: GradNorm: gradient normalization for adaptive loss balancing in deep multitask networks. arXiv preprint arXiv:1711.02257 (2017)

3. Collobert, R., Weston, J.: A unified architecture for natural language processing: deep neural networks with multitask learning. In: Proceedings of the 25th International Conference on Machine Learning, pp. 160–167. ACM (2008)
4. Conneau, A., Kiela, D., Schwenk, H., Barrault, L., Bordes, A.: Supervised learning of universal sentence representations from natural language inference data. arXiv preprint arXiv:1705.02364 (2017)
5. Dai, A.M., Le, Q.V.: Semi-supervised sequence learning. In: Advances in Neural Information Processing Systems, pp. 3079–3087 (2015)
6. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
7. Howard, J., Ruder, S.: Universal language model fine-tuning for text classification. arXiv preprint arXiv:1801.06146 (2018)
8. Kiros, R., et al.: Skip-thought vectors. In: Advances in Neural Information Processing Systems, pp. 3294–3302 (2015)
9. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: International Conference on Machine Learning, pp. 1188–1196 (2014)
10. Lin, Z., et al.: A structured self-attentive sentence embedding. arXiv preprint arXiv:1703.03130 (2017)
11. Liu, L., et al.: Empower sequence labeling with task-aware neural language model. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
12. Liu, X., Gao, J., He, X., Deng, L., Duh, K., Wang, Y.Y.: Representation learning using multi-task deep neural networks for semantic classification and information retrieval (2015)
13. Liu, X., He, P., Chen, W., Gao, J.: Multi-task deep neural networks for natural language understanding. arXiv preprint arXiv:1901.11504 (2019)
14. Logeswaran, L., Lee, H.: An efficient framework for learning sentence representations. arXiv preprint arXiv:1803.02893 (2018)
15. Maas, A.L., Daly, R.E., Pham, P.T., Huang, D., Ng, A.Y., Potts, C.: Learning word vectors for sentiment analysis. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1, pp. 142–150. Association for Computational Linguistics (2011)
16. McCann, B., Bradbury, J., Xiong, C., Socher, R.: Learned in translation: contextualized word vectors. In: Advances in Neural Information Processing Systems, pp. 6294–6305 (2017)
17. McCloskey, M., Cohen, N.J.: Catastrophic interference in connectionist networks: the sequential learning problem. In: Psychology of Learning and Motivation, vol. 24, pp. 109–165. Elsevier (1989)
18. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013)
19. Pennington, J., Socher, R., Manning, C.: Glove: global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543 (2014)
20. Peters, M.E., et al.: Deep contextualized word representations. arXiv preprint arXiv:1802.05365 (2018)
21. Qiao, C., et al.: A new method of region embedding for text classification. In: International Conference on Learning Representations (2018)

22. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I.: Improving language understanding by generative pre-training (2018). https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/languageunsupervised/languageunderstanding paper.pdf
23. Rei, M.: Semi-supervised multitask learning for sequence labeling. arXiv preprint arXiv:1704.07156 (2017)
24. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, pp. 5998–6008 (2017)
25. Voorhees, E.M., Tice, D.M.: The TREC-8 question answering track evaluation. In: TREC, vol. 1999, p. 82. Citeseer (1999)
26. Wu, Y., et al.: Google's neural machine translation system: Bridging the gap between human and machine translation. arXiv preprint arXiv:1609.08144 (2016)
27. Yosinski, J., Clune, J., Bengio, Y., Lipson, H.: How transferable are features in deep neural networks? In: Advances in Neural Information Processing Systems, pp. 3320–3328 (2014)
28. Zhang, X., Zhao, J., LeCun, Y.: Character-level convolutional networks for text classification. In: Advances in Neural Information Processing Systems, pp. 649–657 (2015)