Fusing Label Embedding into BERT: An Efficient Improvement for Text Classification

Yijin Xiong, Yukun Feng, Hao Wu, Hidetaka Kamigaito, and Manabu Okumura
Institute of Innovative Research, Tokyo Institute of Technology
{yijinx,yukun,wuhao,kamigaito,oku}@lr.pi.titech.ac.jp

Abstract

With pre-trained models, such as BERT, gaining more and more attention, plenty of research has been done to further promote their capabilities, from enhancing the experimental procedures (Sun et al., 2019) to improving the mathematical principles. In this paper, we propose a concise method for improving BERT's performance in text classification by utilizing a label embedding technique while keeping almost the same computational cost. Experimental results on six text classification benchmark datasets demonstrate its effectiveness.

1 Introduction

Text classification is a classic problem in natural language processing (NLP). The task is to annotate a predefined class or classes to a given text, where text representation is an important intermediate step.

A variety of neural models have been developed to learn better text representations, including convolution models (Kim, 2014; Kalchbrenner et al., 2014; Zhang et al., 2015; Conneau et al., 2017; Johnson and Zhang, 2017; Zhang et al., 2017; Shen et al., 2018), recurrent models (Liu et al., 2016; Yogatama et al., 2017; Seo et al., 2017; Wang et al., 2018b), and attention mechanisms (Yang et al., 2016; Lin et al., 2017).

Pre-trained models have also been greatly beneficial in text classification in that they help streamline the training process by avoiding a start from zero (Stein et al., 2019; Wang et al., 2017; Jiang et al., 2019). One group of approaches has focused on word embeddings, such as word2vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014); another has focused on contextualized word embeddings, from CoVe (McCann et al., 2017) to ELMo (Peters et al., 2018), OpenAI GPT (Radford et al., 2018), ULMFiT (Howard and Ruder, 2018), and BERT (Devlin et al., 2019).

BERT has achieved particularly impressive performances across a variety of NLP tasks. With its success, models pre-trained on a large amount of data, such as ERNIE (Zhang et al., 2019), RoBERTa (Liu et al., 2019), UniLM (Dong et al., 2019), and XLnet (Yang et al., 2019), have become popular thanks to their ability in learning contextualized representations. These models are based on the multi-layered bidirectional attention mechanism (Vaswani et al., 2017) and are trained through the masked word prediction task, which are two of the main components of BERT. Continuing to investigate the potential of BERT remains important, since the findings can help with the investigation of variants of BERT as well.

In this work, we propose a simple but effective method to improve BERT's performance in text classification. We enhance the contextual representation learning through encoding the texts of class labels (e.g. "world", "sports", "business", and "science technology" in the AGNews dataset) along with the documents, without changing the original encoder structure. Our main contributions are as follows.

- The embeddings of both texts and labels are jointly learned from the same latent space, and so no further intermediate steps are needed.
- Our implementation takes more thorough and efficient advantage of BERT's inherent self-attention for the interaction between the label embeddings and text embeddings, without introducing other mechanisms.
- Since only the original structure of BERT is required, our method barely increases the amount of computation.
- Extensive results on six benchmark datasets reveal that our method taps into the deeper

Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pages 1743–1750

potential of BERT, leading to optimism that BERT can be further improved for text classification as well as other downstream tasks.

2 Related Work

Apart from the pre-trained models for learning general language representations mentioned above, some studies have focused specifically on leveraging the representations of classes or the higher level global information. Examples include t-BERT (Peinelt et al., 2020), which combines topic models with BERT for pairwise semantic similarity detection, and LCM (Guo et al., 2020), which generates an enhancement distribution to the one-hot vector representing the classes by calculating the similarity between instances and labels to improve the classification performance.

Moreover, the label embedding has increasingly taken a leading role in related research. It is a technique in which the contents of labels are also embedded, so that the model can be trained to deal with the label information and input features at the same time. It is proven to be effective in various domains including image classification (Akata et al., 2015), multi-modal learning between images and texts (Frome et al., 2013; Kiros et al., 2014), text recognition in images (Rodriguez-Serrano et al., 2013), and zero-shot learning (Palatucci et al., 2009; Yogatama et al., 2015; Li et al., 2015; Ma et al., 2016).

Notably, in the field of text classification, Zhang et al. (2018) converted the task into a vector-matching problem, while Yang et al. (2018) utilized a sequence generation framework for capturing the correlation between labels. Wang et al. (2018a) proposed the label embedding attentive model (LEAM), an attention-based framework that jointly learns the embeddings of words and labels from a shared latent space. Inspired by LEAM, Si et al. (2020) developed LESA-BERT, where label embeddings are incorporated into self-attention by modifying attention scores. Our approach differs from them in that it can consider bidirectional attention between both label and document embeddings in BERT without changing its attention process.

3 Method

3.1 Fusing Label Embedding into BERT

Figure 1 shows the network structure of our model. Inspired by the sentence pair input configuration of

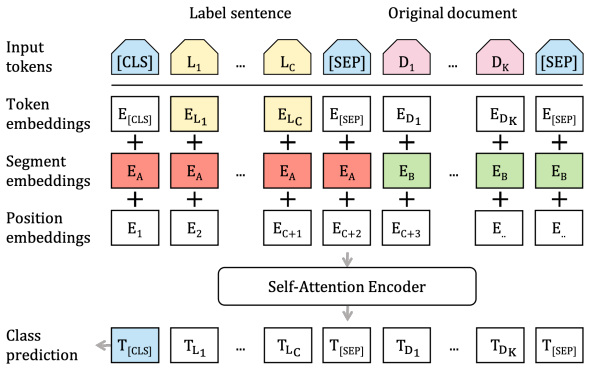

Figure 1: Structure of proposed method.

BERT, we concatenate texts of labels and an original document to be classified with a [SEP] token as an input, and use different segment embeddings for the label texts and the document text. The actual label texts are listed in Appendix A.

We denote the document tokens as D_i and their corresponding token embeddings as E_{D_i}. Hence, D_K refers to the last token of the input document, where K is the number of words in the document. Let L_j be the label texts of the j-th class of the total C classes. Since L_j may consist of several subwords, we calculate E_{L_j}, the embedding of L_j, by averaging the token embeddings of all subwords in L_j. In this way, the length of the label sentence is equal to C, and E_{L_j} can be encoded together with E_{D_i} through self-attention. We denote this method as w/ [SEP].

Then, following the same process as the original BERT, we apply a linear layer with the Tanh activation function to the last layer of the hidden-state at the [CLS] token, $T_{[CLS]}$, for making the input of the softmax layer. We use cross-entropy loss for the training.

In addition to the paired input, we examine another setting that concatenates label texts and a document text without utilizing [SEP] or discriminating their segment embeddings. The procedure of computing the token embeddings stays consistent with the paired input setting. We denote this method as w/o [SEP].

3.2 Further Enhancement Using tf-idf

In addition to encoding the original texts of labels into BERT with the document, we experiment with selecting more words as representatives for each class, which expands the number of tokens in L_j. We investigate whether this enhancement can further improve the performance of our models. After

tokenizing all the documents under one class in the training set by using the Bert Tokenizer based on WordPiece (Wu et al., 2016), we calculate the average tf-idf score of each subword and add the top 5, 10, 15, or 20 as the supplemental label texts to the corresponding class.

4 Experiments

4.1 Datasets

To evaluate the effectiveness of our method, we performed experiments on six benchmark datasets. As the original benchmarks do not include the development set, we randomly created it from the training set (after removing duplicate samples) for each dataset in accordance with the class distribution of the original test set.

We introduce the original size of each dataset below; see Table 1 for detailed statistics of our training, development, and test sets. Except for IMDb, all the datasets we used were originally constructed by Zhang et al. (2015).

- **AGNews** A news article dataset with titles and descriptions, containing 120,000 training samples and 7600 for testing. Four classes are included: World, Sports, Business, and Science & Technology.
- **DBPedia** An ontology classification over 14 classes, containing 560,000 samples for training and 70,000 for testing.
- **Yahoo! Answers Topic** A dataset containing 1,400,000 training samples and 60,000 testing samples with ten categories. Each sample includes the question title, question content, and best answer.
- **IMDb** (Maas et al., 2011) A binary sentiment classification dataset containing 25,000 highly polar movie reviews for training, and 25,000 for testing. Since its training and test sets are originally of the same size, we merged them together and randomly split it into approximately 8:1:1 for training, development, and testing.
- **Yelp Review Full** A dataset extracted from Yelp Dataset Challenge 2015 data by randomly taking 130,000 training samples and 10,000 testing samples for each starred review from 1 to 5. In total, there are 650,000 training samples and 50,000 testing samples.
- **Yelp Review Polarity** A dataset also extracted from Yelp Dataset Challenge 2015 data but coarsely divided into two classes, considering 1 and 2 stars as negative, and 4 and 5 as positive. In total, there are 560,000 training samples and 38,000 testing samples.

4.2 Settings

For both the baselines (BERT and LESA-BERT) and our proposed methods, we used the pre-trained uncased BERT-base model (Wolf et al., 2019), which consists of 12 Transformer blocks (Vaswani et al., 2017) with 12 self-attention heads and the hidden size of 768. We set the learning rate to 2e-5 and the batch size to 24. The drop-out probability was kept at 0.1. For optimization, we used AdamW (Loshchilov and Hutter, 2018) with epsilon of 1e-8.

The models were trained for five epochs for each benchmark. At the end of each epoch, they were evaluated on the development set, and the ones with the highest accuracy were saved. We report those models' performance on the test set. The training was done for AGNews and DBPedia on 2080Ti and for the rest on Titan RTX. See Table 1 for the maximum sentence length and warm-up steps we assigned for each dataset. We decided the max length based on the average length statistics from Sun et al. (2019) to fully utilize the GPU memory.

Note that we used adjectives "bad, poor, fair, good, excellent", representing the number of stars, instead of numbers 1 to 5 for the basic label texts in the Yelp Review Full dataset, since numbers are used in various unrelated contexts, that may lead to ambiguity.

We fixed the number of top-ranked subwords added for each method on each dataset on the development set. For example, Table 3 shows the averaged results on the AGNews development set for the three methods with top-5, 10, 15, and 20 words added. LESA-BERT (Si et al., 2020), w/ [SEP], and w/o [SEP] all reach the highest accuracy when five words were added, and so this was their final configuration when tested. The comparative experiments were also conducted on the other five datasets (see Appendix B for details).

4.3 Experimental Results

In Table 2, we report the average performance with three different random seeds (see Appendix B for

Dataset	Classes	Type	Train	Dev.	Test	Max length	Warm-up steps
AGNews	4	Topic	112,312	7,600	7,600	230	1,000
DBPedia	14	Topic	489,630	70,000	70,000	230	4,300
Yahoo	10	Topic	1,339,933	60,000	60,000	480	11,900
IMDb	2	Sentiment	39,576	4,800	4,800	480	350
Yelp F.	5	Sentiment	599,960	50,000	50,000	480	5,300
Yelp. P	2	Sentiment	521,985	38,000	38,000	480	4,600

Table 1: Statistics of six benchmarks. In each dataset, the development set is of the same size and class distribution as the test set. Max length indicates the text length without label sentences: the total sentence length for w/ [SEP] would be Max Length + C + 1, where C denotes the number of classes. As for w/o [SEP], the length would be Max Length + C.

Model	AGNews	DBPedia	Yahoo	IMDb	Yelp F.	Yelp P.
BERT	94.456	99.123	75.534	94.667	68.334	97.071
LESA-BERT$^\diamond$	94.522	99.164	75.431	94.743	68.411	97.083
Ours w/ [SEP]	94.557	99.147	75.484	94.931	68.605	97.106
Ours w/o [SEP]	94.653	**99.177***	75.494	94.875	**68.651***	97.155
LESA-BERT$^\diamond$ + tf-idf	94.561 (+5)	99.127 (+10)	75.557 (+15)	94.757 (+20)	68.245 (+10)	97.078 (+15)
Ours w/ [SEP] + tf-idf	94.697 (+5)	99.141 (+10)	75.589 (+15)	94.917 (+ 5)	68.367 (+20)	97.165 (+15)
Ours w/o [SEP] + tf-idf	**94.886***(+5)	99.139 (+20)	**75.628** (+15)	**94.938** (+15)	68.252 (+15)	**97.176** (+15)

Table 2: Model accuracy on the test set, in percentage. $^\diamond$We ran LESA-BERT using the authors' implementation. +tf-idf means top-ranked subwords with average tf-idf scores are added for each class as supplemental label texts, and (+k) denotes their number. **Bold** indicates the best score for each dataset. * means the difference from BERT is statistically significant using paired-bootstrap-resampling test with $p<0.05$.

No. of words	+5	+10	+15	+20
LESA-BERT	**94.956**	94.903	94.912	94.903
Ours w/ [SEP]	**94.860**	94.812	94.807	94.802
Ours w/o [SEP]	**94.916**	94.785	94.912	94.846

Table 3: Model performance on the AGNews development set with different numbers of supplemental subwords added.

detailed results). We find that fusing only original label texts either with or without [SEP] yielded an improvement over the baselines, except on Yahoo. We assume this is because the original labels are not discriminative enough for big datasets, and so they may corrupt the input rather than enhance it, that leads to the degradation in accuracy.

However, when the top-ranked words were added, the performance on Yahoo was boosted to exceed the baselines. We notice this improvement, caused by adding supplemental words, took place on most benchmarks. Please note that the added words can sometimes contribute to the performance improvement even for the baseline, LESA-BERT.

On the other hand, the performances of all methods dropped drastically on Yelp F.. We assume this is because the top-ranked subwords with averaged tf-idf scores may not be a good representative for the granularity and polarity of emotions, while they can be powerful enough for distinguishing between topics. The enhancement helped IMDb and Yelp P. but not Yelp F., though all are benchmarks for sentiment analysis. In contrast to IMDb and Yelp P., which have only positive and negative labels, Yelp F. has inherent labels, decided by contexts, and so the effect of the tf-idf-based enhancement might be restricted on Yelp F. because the tf-idf score represents only the importance of the words.

Note that w/o [SEP] is better than w/ [SEP] in most cases. The Next Sentence Prediction (NSP) task, used in BERT to learn sentence-level representations, concatenates two natural language sentences with a [SEP] token. On the other hand, when we concatenate a label sequence with an input document, the [SEP] token combines a non-natural language sequence with a natural language sentence. This difference may have caused the skewness between pre-training and fine-tuning in BERT, leading to the performance degradation. Thus, simply adding a label sequence as a prefix, as in the w/o [SEP] method, which provides information gain, could yield a more stabilized improvement.

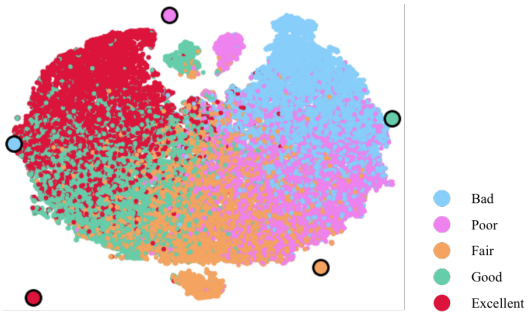

Figure 2: t-SNE visualization of T_{CLS} vectors and averaged T_{L_j} vectors over the Yelp F. test set.

Next, we used t-SNE (Maaten and Hinton, 2008) to visualize the learned representations on a 2-dimensional map, as shown in Figure 2. We visualize the vectors learned from the w/o [SEP] model for the Yelp F. test set. Each color represents a different class. The point clouds are T_{CLS} vectors, and each point corresponds to a test sample. The large dots with black circles are the averaged vectors of T_{L_j}, which is the encoded embedding of each label. Compared with the embedding of [CLS], the label embeddings are more separated in the vector space. This is presumably the reason that the label embeddings can support classification.

5 Conclusion

We proposed a simple but effective method for fusing label embeddings into BERT while utilizing its inherent inputting structure and self-attention mechanism, which leads to having significant improvements on benchmarks of relatively small and medium sizes. The results from the experiments adding subwords with top-ranked average tf-idf scores as supplemental label texts demonstrated that our method can generally improve the performance as expected. As there may be more appropriate methods for constructing enhanced representations, we intend to explore this further in future work. We will also examine different ways of uncovering more potential of pre-trained attentional models like BERT.

Acknowledgements

We thank the anonymous reviewers for their helpful discussion on this work and their valuable comments on the previous draft of the paper.

References

Zeynep Akata, Florent Perronnin, Zaid Harchaoui, and Cordelia Schmid. 2015. Label-embedding for image classification. *IEEE transactions on pattern analysis and machine intelligence*, 38(7):1425–1438.

Alexis Conneau, Holger Schwenk, Loïc Barrault, and Yann Lecun. 2017. Very deep convolutional networks for text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1107–1116, Valencia, Spain. Association for Computational Linguistics.

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, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems*, pages 13063–13075.

Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Marc'Aurelio Ranzato, and Tomas Mikolov. 2013. Devise: A deep visual-semantic embedding model. In *Advances in neural information processing systems*, pages 2121–2129.

Biyang Guo, Songqiao Han, Xiao Han, Hailiang Huang, and Ting Lu. 2020. Label confusion learning to enhance text classification models. *arXiv preprint arXiv:2012.04987*.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Ye Jiang, Johann Petrak, Xingyi Song, Kalina Bontcheva, and Diana Maynard. 2019. Team bertha von suttner at semeval-2019 task 4: Hyperpartisan news detection using elmo sentence representation convolutional network. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 840–844.

Rie Johnson and Tong Zhang. 2017. Deep pyramid convolutional neural networks for text categorization. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 562–570.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 655–665, Baltimore, Maryland. Association for Computational Linguistics.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Ryan Kiros, Ruslan Salakhutdinov, and Richard S Zemel. 2014. Unifying visual-semantic embeddings with multimodal neural language models. *arXiv preprint arXiv:1411.2539*.

Xirong Li, Shuai Liao, Weiyu Lan, Xiaoyong Du, and Gang Yang. 2015. Zero-shot image tagging by hierarchical semantic embedding. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 879–882.

Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. *arXiv preprint arXiv:1703.03130*.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent neural network for text classification with multi-task learning. *arXiv preprint arXiv:1605.05101*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2018. Fixing weight decay regularization in adam.

Yukun Ma, Erik Cambria, and Sa Gao. 2016. Label embedding for zero-shot fine-grained named entity typing. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 171–180.

Andrew 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.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605.

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: Contextualized word vectors. In *Advances in neural information processing systems*, pages 6294–6305.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26:3111–3119.

Mark Palatucci, Dean Pomerleau, Geoffrey E Hinton, and Tom M Mitchell. 2009. Zero-shot learning with semantic output codes. *Advances in neural information processing systems*, 22:1410–1418.

Nicole Peinelt, Dong Nguyen, and Maria Liakata. 2020. tbert: Topic models and bert joining forces for semantic similarity detection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7047–7055.

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, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Jose A Rodriguez-Serrano, Florent Perronnin, and France Meylan. 2013. Label embedding for text recognition. In *BMVC*, pages 5–1.

Minjoon Seo, Sewon Min, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Neural speed reading via skim-rnn. *arXiv preprint arXiv:1711.02085*.

Dinghan Shen, Yizhe Zhang, Ricardo Henao, Qinliang Su, and Lawrence Carin. 2018. Deconvolutional latent-variable model for text sequence matching. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Shijing Si, Rui Wang, Jedrek Wosik, Hao Zhang, David Dov, Guoyin Wang, and Lawrence Carin. 2020. Students need more attention: Bert-based attention model for small data with application to automatic patient message triage. In *Machine Learning for Healthcare Conference*, pages 436–456. PMLR.

Roger Alan Stein, Patricia A Jaques, and Joao Francisco Valiati. 2019. An analysis of hierarchical text classification using word embeddings. *Information Sciences*, 471:216–232.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune bert for text classification? In *China National Conference on Chinese Computational Linguistics*, pages 194–206. Springer.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Guoyin Wang, Chunyuan Li, Wenlin Wang, Yizhe Zhang, Dinghan Shen, Xinyuan Zhang, Ricardo Henao, and Lawrence Carin. 2018a. Joint embedding of words and labels for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2321–2331, Melbourne, Australia. Association for Computational Linguistics.

Jin Wang, Zhongyuan Wang, Dawei Zhang, and Jun Yan. 2017. Combining knowledge with deep convolutional neural networks for short text classification. In *IJCAI*, volume 350.

Wenlin Wang, Zhe Gan, Wenqi Wang, Dinghan Shen, Jiaji Huang, Wei Ping, Sanjeev Satheesh, and Lawrence Carin. 2018b. Topic compositional neural language model. In *International Conference on Artificial Intelligence and Statistics*, pages 356–365. PMLR.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Pengcheng Yang, Xu Sun, Wei Li, Shuming Ma, Wei Wu, and Houfeng Wang. 2018. Sgm: sequence generation model for multi-label classification. *arXiv preprint arXiv:1806.04822*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5753–5763.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 1480–1489.

Dani Yogatama, Chris Dyer, Wang Ling, and Phil Blunsom. 2017. Generative and discriminative text classification with recurrent neural networks. *arXiv preprint arXiv:1703.01898*.

Dani Yogatama, Dan Gillick, and Nevena Lazic. 2015. Embedding methods for fine grained entity type classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 291–296.

Honglun Zhang, Liqiang Xiao, Wenqing Chen, Yongkun Wang, and Yaohui Jin. 2018. Multi-task label embedding for text classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4545–4553, Brussels, Belgium. Association for Computational Linguistics.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28:649–657.

Yizhe Zhang, Dinghan Shen, Guoyin Wang, Zhe Gan, Ricardo Henao, and Lawrence Carin. 2017. Deconvolutional paragraph representation learning. In *Advances in Neural Information Processing Systems*, pages 4169–4179.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: Enhanced language representation with informative entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1441–1451, Florence, Italy. Association for Computational Linguistics.

Appendix

A. Original Label Texts

See Table 4 for the basic label texts for each dataset. Except for Yelp F., all the texts are provided by the original constructors of the datasets.

B. Detailed Experimental Results

Tables 5 - 15 are the averaged development and detailed test results for each dataset, respectively. **Bold** indicates the best score for each model on the devlopment set and among the models on the test set. * means the difference from BERT is statistically significant using paired-bootstrap-resampling test with $p<0.05$.

Dataset	Classes	Label Texts
AGNews	4	world, sports, business, science technology
DBPedia	14	company, educational institution, artist, athlete, office holder, mean of transportation, building, natural place, village, animal, plant, album, film, written work
Yahoo	10	society culture, science mathematics, health, education reference, computers internet, sports, business finance, entertainment music, family relationships, politics government
IMDb	2	negative, positive
Yelp F.	5	bad, poor, fair, good, excellent
Yelp P.	2	negative, positive

Table 4: Basic label texts of the six benchmarks.

AGNews	seed 1	seed 2	seed 3	Mean
BERT	94.592	94.421	94.355	94.456
LESA-BERT$^\diamond$	94.671	94.382	94.513	94.522
Ours w/ [SEP]	94.605	94.474	94.592	94.557
Ours w/o [SEP]	94.697	94.645	94.618	94.653
LESA-BERT$^\diamond$ + 5	94.487	94.605	94.592	94.561
Ours w/ [SEP] + 5	**94.947***	94.658	94.487	94.697
Ours w/o [SEP] + 5	94.776	**94.921***	**94.961***	**94.886***

Table 5: Test results of AGNews.

No. of words	+5	+10	+15	+20
LESA-BERT$^\diamond$	99.083	**99.098**	99.086	99.092
Ours w/ [SEP]	99.082	**99.091**	99.085	99.081
Ours w/o [SEP]	99.090	99.081	99.092	**99.094**

Table 6: Averaged dev. results of DBPedia.

DBPedia	seed 1	seed 2	seed 3	Mean
BERT	99.136	99.116	99.117	99.123
LESA-BERT$^\diamond$	99.144	**99.184***	99.164*	99.164
Ours w/ [SEP]	99.149	99.133	99.159*	99.147
Ours w/o [SEP]	**99.179**	99.183*	**99.170***	**99.177***
LESA-BERT$^\diamond$ + 10	99.114	99.127	99.139	99.127
Ours w/ [SEP] + 10	99.103	99.177*	99.144	99.141
Ours w/o [SEP] + 20	99.157	99.110	99.151*	99.139

Table 7: Test results of DBPedia.

No. of words	+5	+10	+15	+20
LESA-BERT$^\diamond$	75.522	75.554	**75.561**	75.546
Ours w/ [SEP]	75.541	75.571	**75.574**	75.554
Ours w/o [SEP]	75.603	75.547	**75.621**	75.566

Table 8: Averaged dev. results of Yahoo.

Yahoo	seed 1	seed 2	seed 3	Mean
BERT	75.637	75.397	75.568	75.534
LESA-BERT$^\diamond$	75.305	75.592*	75.397	75.431
Ours w/ [SEP]	75.470	75.552	75.430	75.484
Ours w/o [SEP]	75.482	75.543	75.458	75.494
LESA-BERT$^\diamond$ + 15	75.717	75.478	75.477	75.557
Ours w/ [SEP] + 15	75.617	**75.658***	75.492	75.589
Ours w/o [SEP] + 15	**75.740**	75.567*	**75.576**	**75.628**

Table 9: Test results of Yahoo.

No. of words	+5	+10	+15	+20
LESA-BERT$^\diamond$	94.604	94.722	94.757	**94.812**
Ours w/ [SEP]	**94.708**	94.695	94.694	94.604
Ours w/o [SEP]	94.646	94.653	**94.910**	94.512

Table 10: Averaged dev. results of IMDb.

IMDb	seed 1	seed 2	seed 3	Mean
BERT	94.708	94.438	94.854	94.667
LESA-BERT$^\diamond$	94.750	94.979*	94.500	94.743
Ours w/ [SEP]	94.583	**95.292***	94.917	94.931
Ours w/o [SEP]	**95.167***	94.646	94.813	94.875
LESA-BERT$^\diamond$ + 20	94.813	94.771	94.688	94.757
Ours w/ [SEP] + 5	95.125*	94.917	94.708	94.917
Ours w/o [SEP] + 15	95.063	94.667	**95.083**	**94.938**

Table 11: Test results of IMDb.

No. of words	+5	+10	+15	+20
LESA-BERT$^\diamond$	68.776	**68.819**	68.780	68.779
Ours w/ [SEP]	68.719	68.734	68.701	**68.777**
Ours w/o [SEP]	68.793	68.733	**68.799**	68.795

Table 12: Averaged dev. results of Yelp F.

Yelp F.	seed 1	seed 2	seed 3	Mean
BERT	68.180	68.432	68.390	68.334
LESA-BERT$^\diamond$	68.570*	68.526	68.136	68.411
Ours w/ [SEP]	68.602*	68.600	68.612	68.605
Ours w/o [SEP]	**68.638***	**68.666**	**68.648***	**68.651***
LESA-BERT$^\diamond$ + 10	68.204	68.264	68.268	68.245
Ours w/ [SEP] + 20	68.300	68.392	68.408	68.367
Ours w/o [SEP] + 15	68.172	68.260	68.324	68.252

Table 13: Test results of Yelp F.

No. of words	+5	+10	+15	+20
LESA-BERT$^\diamond$	97.157	97.164	**97.191**	97.169
Ours w/ [SEP]	97.143	97.181	**97.193**	97.169
Ours w/o [SEP]	97.168	97.151	**97.228**	97.181

Table 14: Averaged dev. results of Yelp P.

Yelp P.	seed 1	seed 2	seed 3	Mean
BERT	97.084	97.037	97.092	97.071
LESA-BERT$^\diamond$	97.155	97.050	97.045	97.083
Ours w/ [SEP]	97.116	97.137	97.066	97.106
Ours w/o [SEP]	**97.179**	97.184*	97.103	97.155
LESA-BERT$^\diamond$ + 15	97.082	97.129	97.024	97.078
Ours w/ [SEP] + 15	97.121	97.179*	**97.195**	97.165
Ours w/o [SEP] + 15	97.153	**97.197***	97.179	**97.176**

Table 15: Test results of Yelp P.