Comparative Analysis of SimCSE for minBERT Optimization with Multiple Downstream Tasks

Stanford CS224N Default Project

Runqiu Zhang
Department of Civil and Environmental Engineering
Stanford University
rqzhang@stanford.edu

Abstract

The anisotropy problem limits the expressiveness of sentence embeddings learned from pretrained large models, and contrastive learning with the SimCSE framework is a simple and effective method to alleviate this problem. In this paper, I introduce the SimCSE framework as a pretraining method for minBERT optimization, and the optimized model achieves an average score of 0.763 in multiple downstream tasks. In comparison with in-domain and cross-domain pretraining with the Masked Language Modeling task, the advantage of the supervised SimCSE approach is further revealed. The result of the ablation study indicates that cross-domain pretraining improves the general performance of the model, and the anisotropy problem of minBERT is shown to be alleviated with SimCSE in the case study.

1 Key Information to include

- Mentor: NA
- External Collaborators (if you have any): NA
- Sharing project: NA

2 Introduction

Sentence embedding is the cornerstone of many natural language processing (NLP) tasks. In the past, studies on building sentence embeddings are mainly through either predicting the neighboring sentences of a given sentence with the distributional hypothesis (Kiros et al., 2015; Hill et al., 2016), or incorporating n-gram embeddings into the idea of word2vec (Pagliardini et al., 2017; Mikolov et al., 2013). BERT (Bidirectional Encoder Representations from Transformers), on the other hand, adopts the transformer architecture and generates sentence embeddings by processing the entire input sequence of tokens through multiple layers of self-attention and feed-forward neural networks (Devlin et al., 2018). These contextually rich embeddings are well-suited for a wide range of NLP tasks, including text classification, question answering, and natural language inference. However, for both pretrained large language models and models trained with tied word embedding matrix, the anisotropy problem has been shown to be widespread in their language representations (Gao et al., 2019; Ethayarajh, 2019). The distribution of word and sentence embeddings over the vector space is direction-dependent and even crammed into a narrow cone-shape space, making a positive correlation between any two arbitrary vectors, greatly impairing the expressiveness of these embeddings and rendering useless metrics such as cosine similarity that directly compare embeddings.

In this project, I introduce the SimCSE (simple contrastive sentence embedding) framework (Gao et al., 2021) to improve the sentence embeddings learned from our minBERT model for multiple downstream tasks, i.e., sentiment classification, paraphrase detection, and semantic textual similarity (STS). Both the supervised and unsupervised approaches of this framework are applied as further

pretraining methods, and their effects are compared to the MLM (masked language modeling) task described in Devlin et al. (2018). The results show that after being pretrained with the supervised SimCSE approach, the model is able to outperform both a directly finetuned version of the model and the version that is pretrained with MLM or unsupervised SimCSE approach, on all three downstream tasks.

To further understand the optimization effect of the SimCSE framework under different task and dataset scenarios, I also perform in-domain pre-training and cross-domain pre-training for a single task, and use MLM as a reference. I find that though the effect of different pretraining methods is task-specific, supervised SimCSE approach can generally beat the unsupervised approach and MLM pretraining. Finally, the ablation experiment and case study demonstrate that cross-domain data are more likely to improve the generalized task performance of the pretrained model, and that the anisotropy problem of the SimCSE pretrained model is alleviated with signs, respectively.

3 Related Work

Since the anisotropy problem was spotted, many researchers have developed different methods to approach the solution. In the original paper that identifies the anisotropic embeddings learned from models trained with tied word embedding matrix by Gao et al. (2019), the authors show that a regularization method can mitigate the representation degeneration problem. Another natural way is post-processing. Li et al. (2020) proposes the BERT-flow method to map the anisotropic sentence embedding distribution from BERT to an isotropic Gaussian distribution, and reaches a new state-of-the-art performance. On top of this, BERT-whitening simplifies the post-processing process into a simple linear transformation, which achieves comparable results with BERT-flow while making the dimensionality reduction operation possible (Su et al., 2021).

Unlike the above methods, contrastive learning aims to bring similar instances closer and push dissimilar ones farther. Based on the work of Gao et al. (2021), the contrastive object has been proven to be effective in alleviating the anisotropy problem. In addition, the SimCSE framework also provides an efficient solution to construct positive pairs, namely semantically close neighbors, for contrastive learning, which is applying the standard dropout twice on intermediate representations of the same sentence for an unsupervised approach, and using the entailment pairs from the natural language inference (NLI) datasets for a supervised approach. This framework not only is simple, but also outperforms previous (more complex) discrete operators for data augmentation such as word deletion, reordering, and substitution (Wu et al., 2020; Meng et al., 2021), so it has a broad application in NLP tasks, especially for the unsupervised approach that does not need additional resources.

However, the success of SimCSM in the original paper is still built on a large amount of (general) training data and emphatically evaluated only on STS tasks without other finetuning, leaving its effect with small within-task datasets in tasks other than STS as an auxiliary step an open question. Our default project fits this study because the three required downstream tasks are the STS task, paraphrase detection that is somewhat related to the STS task, and sentiment classification that is different from the STS task, and all data provided are labeled. By employing the SimCSE framework as a pretraining method and comparing it to the MLM task, we can learn its optimization effect on the minBERT model for both different single-task scenarios and simultaneous multiple tasks, and explore the fundamental factors that bring this optimization.

4 Approach

4.1 Model Architecture Optimization and Baseline

The BERT model itself has been described in detail in the original paper by Devlin et al. (2018) and our default project handout, but how to choose the architecture of the heads for downstream tasks remains a challenging problem.

In this paper, I use a simple brute-force method to experiment with various possible (light) head architectures, including linear layers combined with different activation functions. Due to time constraints, it is not feasible to use grid search to examine all reasonable combinations of layers with different hyperparameters, but through this preliminary experiment, the significant performance gains of my model compared to the initial intuitive implementation, and the closeness to the state-of-the-art

in performance in three tasks demonstrate the effectiveness of this optimization process (details are in the Results section).

I take this optimized model after finetuning on the downstream tasks (but without further pretraining before finetuning) as the *baseline* to show the effect of my pretraining methods. This finetuning is conducted with a straightforward round-robin strategy with three datasets (that are described more in the Data section) to train the model against three objectives simultaneously. Different batch sizes are chosen to make the number of batches from each dataset in the same epoch approximately equal.

4.2 Comparative Analysis of SimCSE Framework

The comparative analysis consists of two parts: 1) the comparison between unsupervised and supervised approaches with different datasets and 2) the comparison between the integration of SimCSE and further pretraining with MLM task, which are both implemented by myself with reference to the original BERT and SimCSE paper. Unlike the approach in the SimCSE paper which trains the model with a large amount (1m) of general data, this project would train the model with the within-task datasets to study the effect of in-domain, cross-domain, and cross-dataset (in sentiment classification) training.

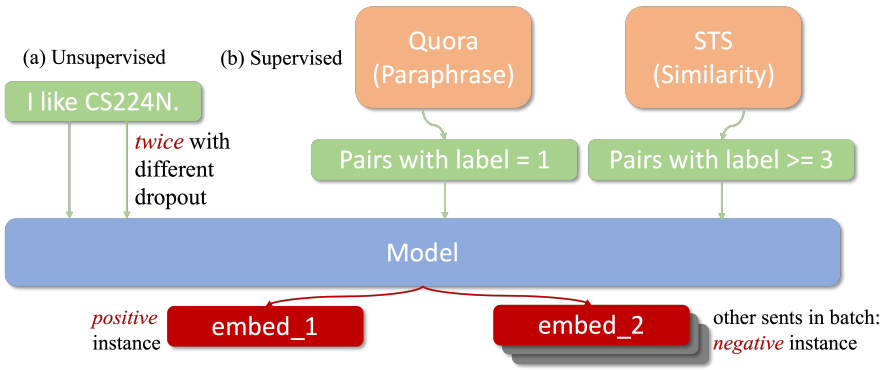

Figure 1: SimCSE framework schematics for the (a) unsupervised approach and (b) supervised approach in this project.

As illustrated in Figure 1, in the unsupervised approach, we obtain two different embeddings for semantically close neighbors by passing the same sentence to our model with the standard dropout *twice*, and use the embeddings as positive pairs. In the supervised approach, the original paper employs the "entailment" pairs from natural language inference (NLI) datasets as positives. However, for our downstream tasks, I think it is reasonable to use the "Is Paraphrase: Yes" question pairs from the Quora dataset and sentence pairs with a similarity label greater than or equal to 3.0 from the SemEval STS Benchmark dataset as positive instances, since both the labels indicate the existence of similarity between the corresponding sentences. For both approaches, the negative instances are simply other sentences within the same mini-batch.

Based on the goal of contrastive learning of keeping semantically related neighbors together and pushing non-neighbors apart for better representation, the training objective to predict the positive pairs (x_i, x_i^+) among negatives with a mini-batch of N pairs is:

$$\ell_i = -log \frac{e^{sim(\mathbf{h}_i, \mathbf{h}_i^+)/\tau}}{\sum_{j=1}^{N} e^{sim(\mathbf{h}_i, \mathbf{h}_i^+)/\tau}} \tag{1}$$

where $\mathbf{h}_i$ and $\mathbf{h}_i^+$ denote the representations of x_i and x_i^+, τ is a temperature hyperparameter, and $\text{sim}(\mathbf{h}_1, \mathbf{h}_2)$ is the cosine similarity $\frac{\mathbf{h}_1^\top \mathbf{h}_2}{\|\mathbf{h}_1\| \cdot \|\mathbf{h}_2\|}$.

5 Experiments

5.1 Data

The datasets used in this project are the Stanford Sentiment Treebank (SST) dataset and the CFIMDB dataset for sentiment classification, the Quora dataset for paraphrase detection, and the SemEval STS Benchmark dataset for semantic textual similarity (STS) task, which have already been described in detail in the handout.

5.2 Evaluation Method

The evaluation metrics used in this default project for downstream tasks are accuracy for sentiment classification and paraphrase detection, and Pearson score for semantic textual similarity. The Pearson score which comes from the original SemEval Paper (Agirre et al., 2013) is calculated based on the true similarity values against the predicted similarity values across the dev or test dataset.

For supervised and unsupervised contrastive learning, I get the [CLS] embeddings from the model for each sentence pair and figure out the cosine similarity values following the SimCSE paper, but adopt the Pearson correlation for the comparison between the similarity values and the STS dataset labels instead of Spearman's correlation in the original paper to maintain consistency with the downstream STS task within the scope of this paper. Additionally, the Pearson correlation is also a commonly used metric in other STS research. For MLM pretraining, the accuracy of predicted word embeddings against the true masked embeddings is calculated.

5.3 Experimental Details

5.3.1 Model Architecture Optimization

The minBERT model is built following the instructions of our handout, which utilizes a WordPiece tokenizer and consists of a trainable embedding layer (where the token embeddings, the segmentation embeddings, and the position embeddings are added), 12 Encoder Transformer layers, and a linear layer with activation function for the [CLS] embedding output.

To find the optimal head architecture for each downstream task, I use the default hyperparameters for training, including 10 epochs, a learning rate of 1e-5, a hidden dropout probability of 0.1 for BERT, a batch size of 32 to fit in the Deep Learning AMI GPU PyTorch 1.12.0 (Ubuntu 20.04) 20220913 (g5.2xlarge). My initial intuitive implementation is adding a linear layer for the sentiment classification to project the [CLS] embedding to logits for 5 classes, calculating the cosine similarity between the two [CLS] embeddings obtained by feeding each sentence of a pair to the model separately as the logit before a sigmoid function for paraphrase detection, and rescaling the cosine similarity obtained in the same way to 0-5 by multiplying 2.5 and adding another 2.5 for the STS task. Of the scores obtained by round-robin multitask training, the STS task has the largest gap from the state-of-the-art, so I decide to experiment with different architectures for the single STS task first, which can significantly save time compared to multitask training. It must be acknowledged that the optimal head architecture for a single task is not necessarily optimal for simultaneous training of multiple tasks, but it is the most feasible approach given the limited time available.

For the STS task, I experiment with the following 9 head architectures ("cos" stands for cosine similarity calculated on two vectors, and "linear" stands for a linear layer): linear -> cos$\times$2.5+2.5, linear -> ReLU(cos$\times$2.5+2.5), linear -> ReLU(cos$\times$5), ReLU(cos$\times$2.5+2.5) only, ReLU(cos$\times$4.5+0.5), ReLU(cos$\times$4.25+0.75), GELU(cos$\times$4.25+0.75), a single linear layer projecting an embedding obtained by feeding BERT a long sentence concatenated by each pair of sentences to a scalar logit, and the single linear layer with a dropout layer before it. The ReLU and GELU functions are chosen based on the thought that cosine similarity values less or equal to 0 all indicate dissimilarity and could probably be zeroed out. Since paraphrase detection is also a task that identifies similarities between sentences, the architecture that performed best in the STS task is directly adapted to paraphrase detection and also shows significant enhancements over the original implementation.

For sentiment classification, the head architectures I experiment with include: linear, ReLU -> linear, GELU -> linear, Tanh -> linear, linear -> ReLU, linear -> GELU, and linear -> Tanh. The

combinations of the top two best architectures in each individual task are then trained by multitasks to determine the best multitasking architecture.

5.3.2 SimCSE Contrastive Learning and MLM Pretraining

For multitask in-domain pretraining, the three datasets are mixed for MLM task and unsupervised SimCSE approach. Each sentence pair from the Quora dataset and the SST dataset is split into two separate pieces of data. The "Is Paraphrase: Yes" question pairs from the Quora dataset and sentence pairs with a similarity label greater than or equal to 3.0 from the STS dataset are mixed for the supervised SimCSE approach. For single-task in-domain pretraining, the SST dataset is used for MLM task and unsupervised SimCSE task before finetuning in the same downstream task, while the STS dataset is used for the MLM task, unsupervised and supervised SimCSE task.

For the cross-domain pretraining, the Quora dataset is used for the MLM task, unsupervised and supervised SimCSE approach pretraining before finetuning in sentiment classification and STS task. In addition, the CFIMDB dataset is also adopted for cross-dataset pretraining with MLM and unsupervised SimCSE tasks in sentiment classification.

Based on experiments with learning rates of 1e-5, 2e-5, and 3e-5, a learning rate of 3e-5 is used for the supervised SimCSE approach with the STS dataset and unsupervised SimCSE approach with all the three single datasets, and 2e-5 for the supervised approach with the Quora or mixed dataset and unsupervised approach with the mixed dataset. A dropout probability of 0.2 or 0.3 corrupts the embeddings excessively through experiments, so I select a probability of 0.1 for all the unsupervised SimCSE pretraining. Since the loss value decreases too quickly for the unsupervised approach, the model evaluation is conducted every 1,000 batches with a size of 32 instead of a whole epoch through the training process with 25,000 batches. The supervised approach uses 10 epochs. Similarly, a learning rate of 3e-5 and a dropout probability of 0.1 are chosen for the MLM pretraining with 10 epochs for single datasets and 20 epochs for mixed dataset.

5.4 Results

5.4.1 Model Architecture Optimization

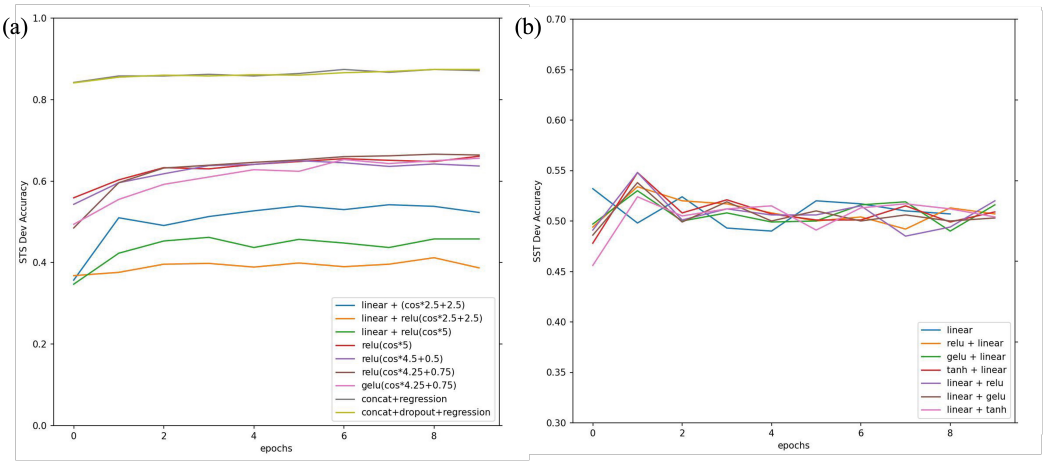

Figure 2: The dev accuracy curves of finetuning for (a) STS head architecture experiments and (b) SST head architecture experiments.

As illustrated in Fig 2 (a), by using different activation functions and removing the linear layer in the head architecture, the accuracy of finetuning in the STS task improves from around 0.4 to beyond 0.6, but concatenating the two sentences and feeding them to BERT as a whole with a single projection linear layer performs well beyond expectation. An accuracy of 0.874 indicates that the minBERT model can understand both the data and the task itself without much additional support. Then the same architecture is directly adapted to the paraphrase detection task, and beats the original implementation again with an accuracy of 0.889.

In the sentiment classification task in Fig 2 (b), the performance gap between different head architectures I experiment with is smaller. Among them, the "Tanh -> linear" architecture and "linear -> ReLU" architecture both achieve the highest accuracy of 0.548, so they are both experimented with in the multitask finetuning scenario combined with the optimal architectures in the other two tasks. The final optimal model architecture from my experiments includes a Tanh function with a subsequent linear layer for sentiment classification and a single linear layer without dropout for paraphrase detection and the STS task. This model achieves an accuracy of 0.518 with the SST dev set, an accuracy of 0.877 with the Quora dev set, and a Pearson score of 0.874 with the STS dev set. These results not only improve significantly from my initial implementation as listed in Table 1, but also approaches the state-of-the-art results for each *single* tasks, which are 0.562 (Brahma, 2018), 0.924 (Baevski et al., 2022), and 0.929 (Jiang et al., 2019), respectively. The model shows the effect of this optimization process and serves as the baseline for the following pretraining methods.

5.4.2 In-domain Pretraining with SimCSE Framework and MLM Task

Model Version	SST Acc	Para Acc	STS Corr	Avg Score
Initial intuitive architecture	0.479	0.524	0.553	0.519
Optimized architecture	0.518	0.877	0.874	0.756
pretrained by MLM	0.496	0.873	0.866	0.745
pretrained by Unsup. SimCSE	0.516	0.875	0.865	0.752
pretrained by Sup. SinCSE	**0.527**	**0.880**	**0.882**	**0.763**

Table 1: The performance of different model versions on the dev sets by multitask finetuning.

As shown in Table 1, the performance of the model pretrained with the supervised SimCSE approach before multitask finetuning outperforms the baseline in all three downstream tasks, and improves the average score by another 0.007. This is my best-ever model. In our test set leaderboard, it ranks 8th with a sentiment classification accuracy of 0.527, a paraphrase detection accuracy of 0.880, an STS correlation of 0.882, and an overall score of 0.763. However, the model pretrained with the unsupervised approach or the MLM task performs worse in the downstream tasks. This gap may be due to the inherent differences in the mixed data from different sets, the hyperparameters of these training methods not all tuned to the optimum, and the generalization performance limitations of the unsupervised SimCSE approach to produce positive instances based on exactly the same sentences, e.g., the model would tend to see sentences of the same length as more semantically similar ones (Wu et al., 2021). Although pretraining gives subsequent finetuning a "first-mover" advantage at the beginning, the continuous updating of the parameters may make this advantage disappear as the finetuning proceeds, especially in the case of potentially conflicting gradients in multitask training.

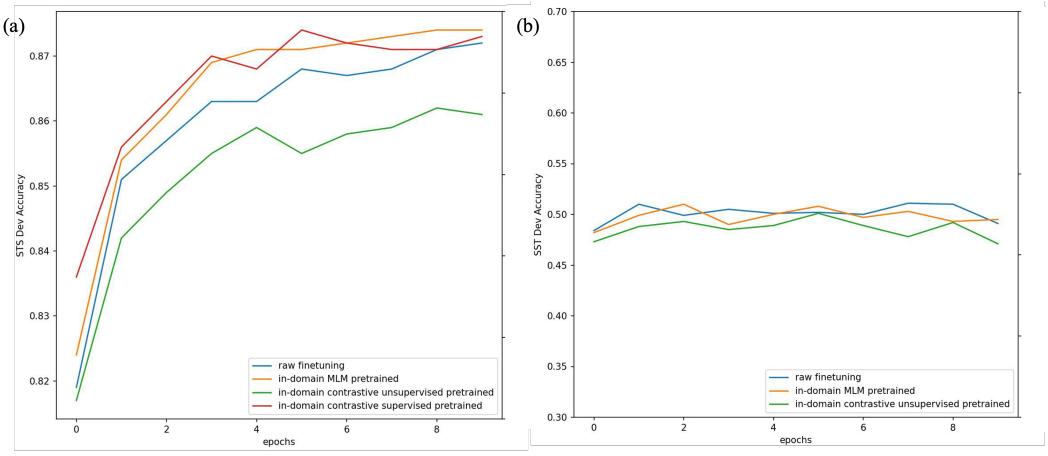

Figure 3: The dev accuracy curves of finetuning following different in-domain pretraining methods in (a) the STS task and (b) the sentiment classification task.

Among the in-domain further pretraining methods for the single STS task in Fig 3 (a), the supervised SimCSE approach achieves the best effect, again indicating the effectiveness of contrastive learning

with labeled positive instances that are similar to each other but not the exact same. In the sentiment classification task in Fig 3 (b), neither of the two unsupervised pretraining methods obtain better results than direct finetuning, which may be related to the small size of the SST dataset.

5.4.3 Cross-domain Pretraining with SimCSE Framework and MLM Task

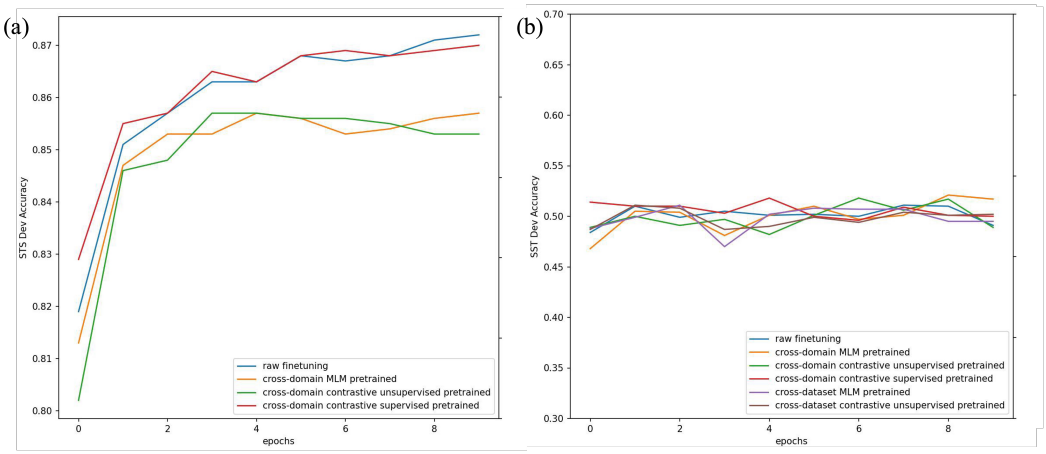

Figure 4: The dev accuracy curves of finetuning following different cross-domain pretraining methods in (a) the STS task and (b) the sentiment classification task.

According to Figure 4, we see that in the STS task, none of the pretraining methods gives a significant improvement to the model, while in the sentiment classification task, the model after pretraining using all three methods performs better than direct finetuning, with MLM pretraining slightly outperforming the other two methods. Comparison between the two tasks suggests that the effect of different pretraining methods is task-specific.

In the sentiment classification task, I also employ cross-dataset pretraining with the CFIMDB dataset. The pretrained model only achieves the same high accuracy as the directly finetuned model, but achieves it much earlier, which validates the first-mover advantage from pretraining mentioned earlier.

6 Analysis

6.1 Ablation Study

Though using the supervised SimCSE approach as a pretraining step offers a performance gain to my model, I am interested in why it works. In particular, the data it uses is derived from my intuition to divide the labels of the Quora and STS datasets, and I am curious about what role each of the two datasets plays in the optimization process. Therefore, I conduct an ablation study to use data from only one of the datasets for pretraining, and the results are as follows:

Datasets Employed	SST Acc	Para Acc	STS Corr	Avg Score
Quora only	0.521	0.878	0.870	0.756
STS only	0.523	0.872	0.875	0.757
Quora + STS	**0.527**	**0.880**	**0.882**	**0.763**
+ MLM objective	0.525	0.875	0.872	0.757

Table 2: The performance of model pretrained with different datasets using the supervised SimCSE approach.

Through this ablation study, we can find that supervised SimCSE pretraining using a single dataset improves the model's performance on the corresponding task in that dataset, but not on the other dataset, compared to the baseline in Table 1. A more generalized model optimization can only be obtained by learning the cross-domain mixed datasets simultaneously. Notably, pretraining using

only one of the two datasets, Quora or STS, both increase the accuracy of the model on the sentiment classification task, which also illustrates the effectiveness of cross-domain pretraining. In addition, I also attempt to incorporate the MLM objective during the pretraining process by multiplying it by a weight of 0.01 and summing it with the contrastive objective, but do not obtain better results. If time permits, more tuning and research on this part should be carried out.

6.2 Anisotropy Case Study

Sentence Pairs	Raw BERT	Finetuned	MLM	Unsup. Sim.	Sup. Sim.
Deep learning World	0.818	0.510	0.845	0.265	0.472
I like you. I love you.	0.993	0.672	0.566	0.866	0.771
I like you. Deep learning is not an easy class.	0.988	-0.497	-0.305	-0.600	-0.139
I can't do it without you. You're the must for me to do it.	0.994	0.757	0.550	0.635	0.766

Table 3: The cosine similarity values for sentence pair samples calculated between the two [CLS] embeddings from different model versions.

To further understand how the SimCSE framework improves the performance of the model, I manually construct several sentence pair samples and feed them into different versions of the model to see whether the anisotropy problem of BERT is alleviated. As shown in Table 3, the relative magnitudes between the values obtained from cosine similarity calculation on the sentence embeddings generated by the original BERT model are basically in line with human perceptions of the similarity of these samples, i.e., the two sentences of samples 2 and 4 are semantically close to each other, while the two sentences of samples 1 and 3 differ more. However, these values are too close in absolute magnitude, proving that the anisotropy problem does exist. Compared to the original BERT, the model directly finetuned on multitasks and the model pretrained with the MLM task and then finetuned both capture the dissimilarity of sample 3, but the distinguishability of their cosine similarity results for samples 1 and 2 is small, and the MLM pretrained model even considers sample 1 more similar than 2. In contrast, the model that is pretrained by the SimCSE framework before finetuning can distinguish the difference in similarity between samples 1 and 2 using cosine similarity as the metric, and the results for the other two samples are reasonable as well. Although the case study cannot mathematically or comprehensively describe the improvement effect of the SimCSE framework on the anisotropy problem of BERT, it is sufficient to illustrate that the difference in sentence embedding expressiveness is one of the candidate reasons why SimCSE pretraining is able to increase model accuracy.

7 Conclusion

In this paper, I introduce the SimCSE framework as a pretraining method to optimize the BERT model and study the effect of the framework in comparison with pretraining through the MLM task. The results show that this framework, especially the supervised learning approach in the SimCSE framework, can improve the model performance. Further in-domain and cross-domain pretraining experiments illustrate that while the effects of different pretraining methods are task-specific, the supervised SimCSE approach tends to outperform the unsupervised SimCSE approach and MLM pretraining, which is in line with my expectation. Based on the ablation study and the case study, I argue that cross-domain pretraining can better improve the generalized task performance of the model, and the SimCSE framework shows signs of alleviating the anisotropy problem of BERT. However, it is worth stating that although my model achieves an average score of 0.763 in the three downstream tasks, the hyperparameter tuning does not strictly follow the grid search approach due to time constraints, and there is room for optimization in the data processing and the experiment setup. In future work, a more quantitative analysis of the anisotropy problem should be performed, for example, by using alignment and uniformity identified in Wang and Isola (2020).

References

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. * sem 2013 shared task: Semantic textual similarity. In *Second joint conference on lexical and computational semantics (* SEM), volume 1: proceedings of the Main conference and the shared task: semantic textual similarity*, pages 32–43.

Alexei Baevski, Wei-Ning Hsu, Qiantong Xu, Arun Babu, Jiatao Gu, and Michael Auli. 2022. Data2vec: A general framework for self-supervised learning in speech, vision and language. In *International Conference on Machine Learning*, pages 1298–1312. PMLR.

Siddhartha Brahma. 2018. Improved sentence modeling using suffix bidirectional lstm. *arXiv preprint arXiv:1805.07340*.

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

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? comparing the geometry of bert, elmo, and gpt-2 embeddings. *arXiv preprint arXiv:1909.00512*.

Jun Gao, Di He, Xu Tan, Tao Qin, Liwei Wang, and Tie-Yan Liu. 2019. Representation degeneration problem in training natural language generation models. *arXiv preprint arXiv:1907.12009*.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*.

Felix Hill, Kyunghyun Cho, and Anna Korhonen. 2016. Learning distributed representations of sentences from unlabelled data. *arXiv preprint arXiv:1602.03483*.

Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Tuo Zhao. 2019. Smart: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization. *arXiv preprint arXiv:1911.03437*.

Ryan Kiros, Yukun Zhu, Russ R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. *Advances in neural information processing systems*, 28.

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. On the sentence embeddings from pre-trained language models. *arXiv preprint arXiv:2011.05864*.

Yu Meng, Chenyan Xiong, Payal Bajaj, Paul Bennett, Jiawei Han, Xia Song, et al. 2021. Coco-lm: Correcting and contrasting text sequences for language model pretraining. *Advances in Neural Information Processing Systems*, 34:23102–23114.

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.

Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. 2017. Unsupervised learning of sentence embeddings using compositional n-gram features. *arXiv preprint arXiv:1703.02507*.

Jianlin Su, Jiarun Cao, Weijie Liu, and Yangyiwen Ou. 2021. Whitening sentence representations for better semantics and faster retrieval. *arXiv preprint arXiv:2103.15316*.

Tongzhou Wang and Phillip Isola. 2020. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, pages 9929–9939. PMLR.

Xing Wu, Chaochen Gao, Liangjun Zang, Jizhong Han, Zhongyuan Wang, and Songlin Hu. 2021. Esimcse: Enhanced sample building method for contrastive learning of unsupervised sentence embedding. *arXiv preprint arXiv:2109.04380*.

Zhuofeng Wu, Sinong Wang, Jiatao Gu, Madian Khabsa, Fei Sun, and Hao Ma. 2020. Clear: Contrastive learning for sentence representation. *arXiv preprint arXiv:2012.15466*.