Exploring Contrastive learning through data augmentation

Hanqing Li
The University of Hong Kong
3036032736
ethanlii@

Chengxiao Li
The University of Hong Kong
3036034564
u3603456@

Yanqiao Li
The University of Hong Kong
3036032750
u3603275@

Wenbin Zhou
The University of Hong Kong
3036032669
zhouwb@

Chao Fan
The University of Hong Kong
3036032970
chaoffan@

Abstract

Learning high-quality sentence representations is beneficial for a wide range of NLP tasks. In recent years, there are a number of Bert fine-tuning-based approaches that have achieved high performance in many downstream tasks. In our reproduction, we select the ACL2021 paper ConSERT as baseline, which presents a contrastive framework for self-supervised sentence representation transfer. We first reproduce the results of their unsupervised setup, and then we set up more data augmentation modules to explore their impact on contrastive learning. To further improve the model performance, we also add a projection head module before the contrastive loss calculation module and provide an early stopping mechanism to prevent the model from "reverse collapse". All code used in this study is available on Github[1].

1 Introduction

Sentence representation learning plays a crucial role in natural language processing tasks (Kiros et al., 2015; Hill et al., 2016; Conneau et al., 2017; Cer et al., 2018). Good sentence representations are beneficial for a wide range of downstream tasks. Recently, BERT-based pre-trained language models have achieved high performance on many downstream tasks.

However, when the BERT-based sentence representations are directly used for the semantic text similarity (STS) task, almost all sentences are mapped to a small region and resulting in high similarity, even though some of these pairs are considered irrelevant by human annotators (Reimers and Gurevych, 2019; Li et al., 2020).

To address the sentence representation problem, ConSERT (Yan et al., 2021) propose a novel sentence-level training objective based on contrastive learning(He et al., 2020; Chen et al., 2020a; Chen et al., 2020b), as shown in Figure 1, it adds multiple data augmentation methods to Bert's embedding layer, generates two different embedding views for each sentence, and achieves Bert fine-tuning by maximizing agreement between representations from same sentence and vise versa. The data augmentation methods proposed in the paper for unsupervised learning include token shuffling, cutoff (Shen et al., 2020) and dropout (Hinton et al., 2012), and it is demonstrated in the experiments that the use of these methods to generate different views is very effective for training sentence representations using contrastive learning.

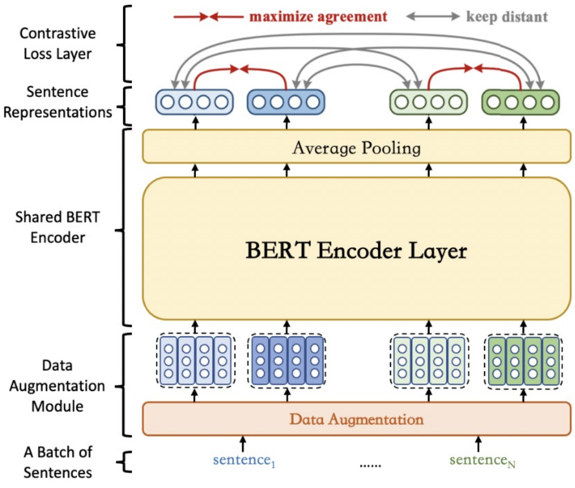


Figure 1: The general framework of ConSERT.

In our work, we reproduce all the unsupervised experiments in the original paper, and inspired by SimCSE (Gao et al., 2021), we believe that adding data augmentation methods to the embedding level alone does not fully exploit the potential of comparative learning, so we mainly make three modifications to the data augmentation: 1. Adding another embedding level data augmentation method: span cutoff ; 2. Adding dropout within the Bert model; 3. Adding the sentence augmentation module (Edunov et al., 2018) through a machine trans-

[1] https://github.com/HanQingLi1228/ECLDA

lation pretrained model to generate high similarity sample for the origin sentence using back translation. After conducting experiments, span cutoff does not work well due to the overcutting of features, but both back translation at the sentence level and dropout within Bert achieve accuracy improvement.

After investigating the current mainstream comparative learning methods in the CV and NLP fields (Chen and He, 2021; Chen et al., 2020a; Gao et al., 2021), we decided to add a projection head to the model to improve the representation quality of the model. We designed our projection head based on SimCLR (Chen et al., 2020a) and a certain degree of improvement has been achieved in the results.

During our research, we found that if we let the ConSERT model train more than a few epochs, the effect would drop extremely fast. To address this issue, we introduced the patience variable, the model is evaluated frequently using STS-Benchmark as the dev set, and the optimal step is retained.

Our contributions can be summarized as follows:

1) We propose sentence level data augmentation method, which improve the effect of contrastive learning.

2) We optimize the original embedding level augmentation combination and propose a span cutoff method.

3) We add the projection head module to ConSERT's training phase, which achieve a result improvement.

4) We developed early stop strategy, prevent the model from "reverse collapsing".

5) We optimized the hyperparameter settings of the original paper to allow the model to achieve better results.

2 Related Works

2.1 Sentence Representation Learning

Supervised Sentence Representation Learning Several works use supervised datasets for sentence representation learning. The supervised Natural Language Inference (NLI) task is found useful to train good sentence representations by (Conneau et al., 2017). They use a BiLSTM-based encoder and train it on two NLI datasets. A Transformer-based architecture was used in Universal Sentence Encoder (Cer et al., 2018), which is based on SNIL dataset to augment the unsupervised training.

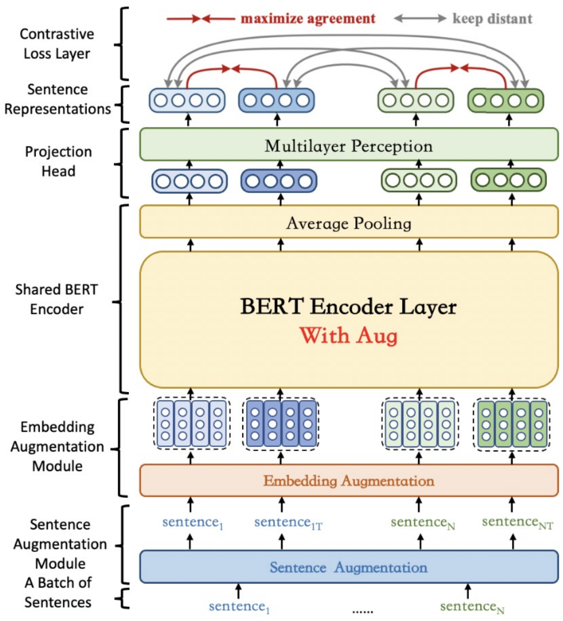


Figure 2: The general framework of our approach.

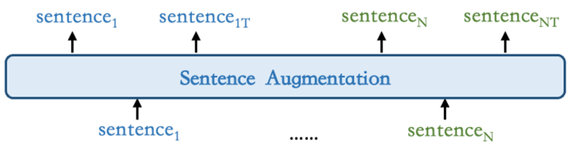


Figure 3: Sentence augmentation.

Self-supervised Pre-training based on Sentence-level As supervised data tagging is costly, researchers are looking for unsupervised training methods. BERT generates a bidirectional Transformer encoder for pre-training in sentence-level, which means it can predict whether two sentences are similar or not. However, the final performance is not so good in namely next sentence prediction. Cross-Thought and CMLM are two similar pertained method, they cut a passage into short sentences and then encode adjacent sentences to restore the masked token in the current sentence. Compared with MLM, additional encoding of other sentences in context is added to help reproduce those masked token, so it is more suitable for sentence-level training. SLM performs self-supervised pre-training by disordering several sentences that are originally coherent, and then by predicting the correct sentence order.

Unsupervised Sentence Representation The pre-training model has been widely used. However, BERT has a worse representation performance in NSP task, and most of us do not have the resources to conduct self-supervised pre-training. Therefore, it is more effective to transfer the representation of the pre-training model to the task. Sim-

CSE (Chen et al., 2021) uses a training framework based on contrastive learning and data augmentation approach with Dropout to fine-tune BERT on Wikipedia corpus.

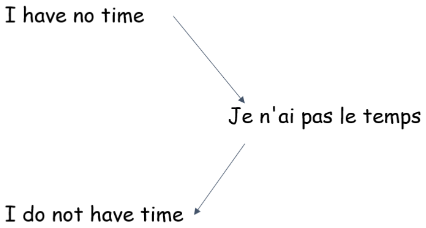


Figure 4: Back translation.

2.2 Contrastive Learning

Contrastive Learning in Computer Vision Recently, unsupervised visual representation learning are more and more popular to use contrastive learning(Chen et al., 2020a; He et al., 2020; Chen et al., 2020b). In this work, They use the normalized temperature-scaled cross-entropy loss (NT-Xent) as the training loss. In the computer vision field, in order to solve the problem without a larger annotated data set, how to adopt the self-supervised pre-training mode to absorb the prior knowledge distribution of the image itself and get a pre-training model.

Contrastive Learning in Natural Language Processing The unsupervised pre-training before natural language processing is all at the word-level. The method of using contrast learning to get better overall image features has inspired the field of natural language processing to learn better sentence representation. Many works use contrastive learning to pre-train language model such as MoCo (He et al., 2020), BERT-CT (Carlsson et al., 2021) and so on. Interestingly, the success of unsupervised learning at the word-level in natural language processing makes the thinking in computer vision field improve the effect of unsupervised learning.

3 Methods

3.1 General Framework

Our framework is mainly based on ConSERT (Gao et al., 2021), by comparing Figure 1 and Figure 2, there are three improvements in terms of framework level:

1) Compare with the origin single embedding augmentation module, we propose a sentencen augmentation module that generates high similarity sentences for input samples.

2) Compare with the naive Bert encoder, we add dropout augmentation inside, and we also add span cutoff as an option of embedding augmentation module.

3) Compare with using the outputs of average pooling directly, we pass them through a multi-layer perceptron , also known as the projection head, to improve the representation quality of the model.

3.2 Sentence Translation

As shown in the Figure 3, in the sentence augmentation part, we use back translation to enhance the sentence, eventually we obtain both original sentence and corresponding translated sentence.

Back translation refers to the process of translating existing sentences in language A into another language B, and then translating back to language A to obtain expanded sentence pairs. As observed in Edunov et al., 2018, back-translation can generate different expressions while preserving the semantics of the original sentence. As shown in the Figure 4, in our work we translated the original English sentence into French and then translated it back to English and finally got the corresponding sentence pair.

In our work, we mainly fine-tune Bert to make similar sentences have similar expressions, and the back-translated data is closer to the real expression than other sentence-level augmentations such as synonym replacement (Keskisärkkä, 2012), so it can better make the model learn similar expressions. However, there are still some noises in the back-translation data obtained by using the translation model, so we consider the back-translation to be weakly supervised.

3.3 Embedding Augmentation

We adopt 5 different embedding level data augmentation strategies to generate views for contrastive learning, including token shuffling, token cutoff, feature cutoff, span cutoff (Shen et al., 2020) and dropout (Hinton et al., 2012), as shown in Figure 5.

Token Shuffling In this strategy, we aim to randomly shuffle the order of the tokens in the input sequences. Since the bag-of-words nature in the transformer architecture, the position encoding is the only factor about the sequential information. Thus, similar to (Lee et al., 2020), we implement

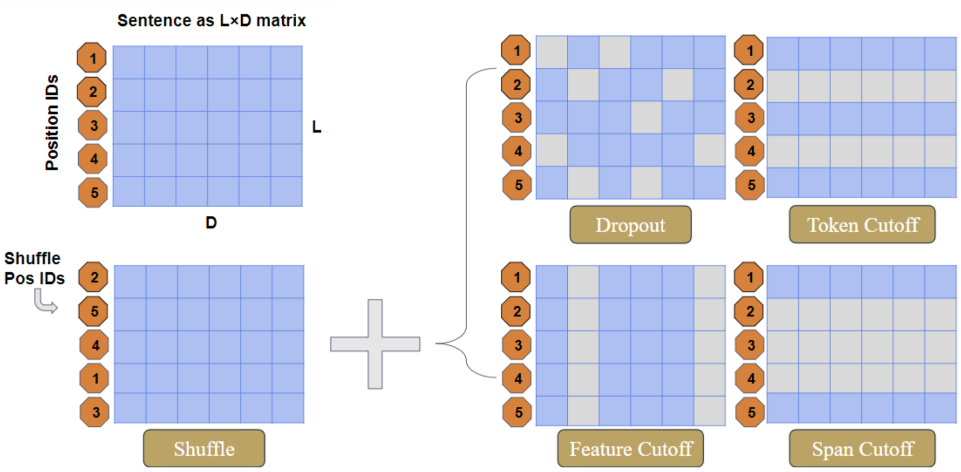


Figure 5: Embedding Augmentation

Method	STS12	STS13	STS14	STS15	STS16	STSb	SICK-R	Avg.
ConSERT-unsup	64.70	78.00	68.60	78.91	75.15	73.59	66.77	72.25
+Embedding aug	65.55	78.86	68.65	80.11	75.82	74.23	68.71	73.13
+Sentence aug	73.09	81.59	72.62	81.04	75.51	77.27	66.95	75.44
+Projection head	74.77	82.45	74.79	81.71	76.52	79.08	68.81	**76.88**
ConSERT-sup	70.92	79.98	74.88	81.76	76.46	78.99	78.15	77.31

Table 1: Experimental results.

this strategy by passing the shuffled position ids to the embedding layer while keeping the order of the token ids unchanged.

Dropout Dropout is a widely used regularization method that avoids overfitting. However, in our experiments, we also show its effectiveness as an augmentation strategy for contrastive learning. For this setting, we randomly drop elements in the token embedding layer by a specific probability and set their values to zero. Note that this strategy is different from Cutoff since each element is considered individually. We activate attention dropout and hidden dropout in Bert to widen difference between representations from sentence during Bert encoding.

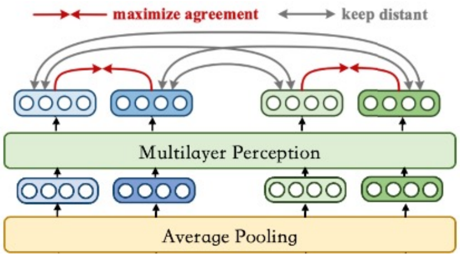


Figure 6: Projection head.

Cutoff (Shen et al., 2020) proposes a simple and efficient data augmentation strategy called cutoff. They randomly erase some tokens (for token cutoff), feature dimensions (for feature cutoff), or token spans (for span cutoff) in the L × d feature matrix.

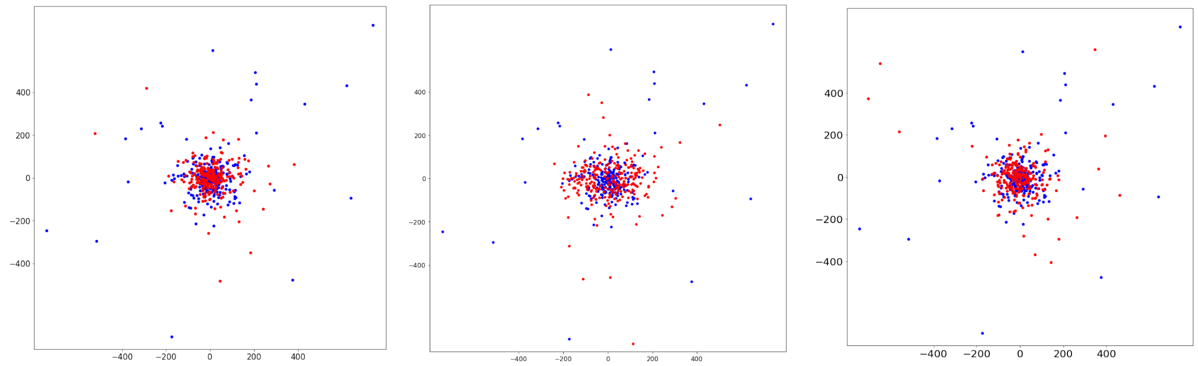
3.4 Projection Head

To add a projection module after encoder and pooling layer, we conduct a non-linear projection head inspired by SimCLR (Chen et al., 2020a). The module maps representations to the space where contrastive loss is applied as shown in Figure 6. The authors of SimCLR find it beneficial to define the contrastive loss on the output after projection head rather than the output after average pooling directly.

In terms of the specific structure, we use a MLP with one hidden layer to obtain the input for loss function, and we select ReLU as the non-linear activation.

3.5 Early Stop Strategy

In contrastive learning process, there will be a model collapse problem. It's like instead of distanc-



(a)Origin data (b)Sentence embedding augmentation (c)Back translation augmentation

Figure 7: Embedded feature extracted from different data augmentation operation dimensionality reducted by TSNE

ing the different data vectors on the hyper-sphere, the model maps them all to the same area of points. In our task, the model will give all of the sentence pairs a higher similarity score than they should have. Furthermore, from our research, we found that if we trained the model without stopping it, the performance of the model will drop extremely fast after several epochs. It's called 'reverse collapse', for the model will give a much lower score similarity to the sentence pairs just as the reverse version of the collapse model.

In order to solve this problem, not like Corneanu's and Zhou's works (Corneanu et al., 2020; Zhou et al., 2020), they check some layers' output inside the network to decide when to stop. We use the STS-Benchmark as the dev set to evaluate the model frequently, and save the model's parameter with the best performance. What's more,we set a patience variable, each evaluation without improvement will decrease this patience variable, and when it comes to zero the training will stop.

4 Experiments

4.1 Implementation Details

The experiment result is shown in the Table 1. Our implementation is based on the ConSERT (Yan et al., 2021). Considering the demand for computing power by the amount of model parameters. We use only the BERT-base for our experiments. All our experiments are done with one 2080Ti. The ratio of token cutoff, feature cutoff and span cutoff is set to 0.1, 0.45 and 0.05 respectively. The ratio of dropout is set to 0.5. The temperature η of NT-Xent loss is set to 0.1. We adopt Adam optimizer and set the learning rate to 5e-7. We use a linear learning rate warm-up over 10% of the training steps. The batch size is set to 96 in most of our experiments. We further discuss the influence of the batch size and the temperature in the subsequent section.

4.2 Sentence Augmentation

In the experimental part of sentence augmentation, we use the STS Benchmark dataset for data augmentation. The method of data augmentation is back-translation. We mainly use Baidu Translate[2] for back translation, translating the sentences in STS Benchmark into French and then back to English, and finally 11498 sentence pairs is obtained. The specific comparison results of the data sets are shown in the Table 2. We can see that the back-translated dataset is much smaller than the supervised and unsupervised datasets in ConSERT (Yan et al., 2021).

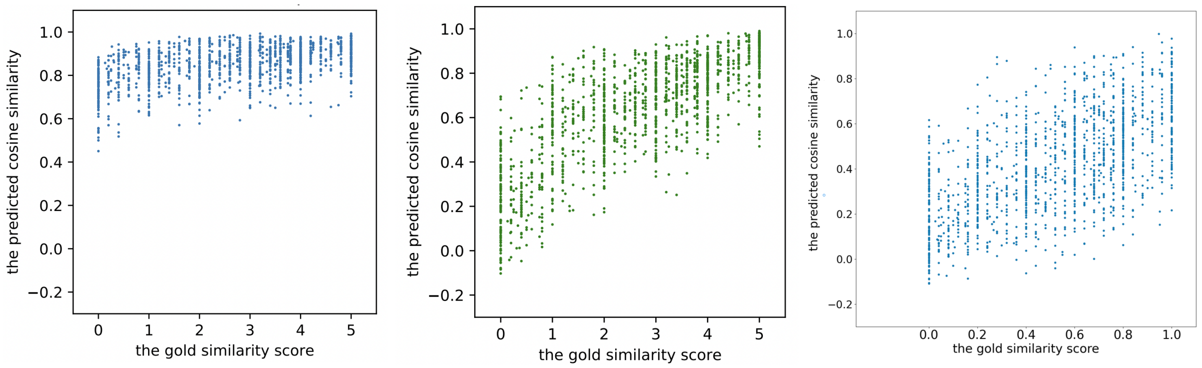
4.3 Embedding Augmentation

As there are three different cutoff strategies, in ConSERT experiments, they only test token cutoff and feature cutoff, with fixed cutoff ratio set to 0.15 and 0.2 respectively, as suggested in (Shen et al., 2020). In our experiment, span cutoff is also considered. We leverage grid search method to find the best dropout/cutoff rate and the best combinations of five embedding augmentation strategies. Our experiments on embedding augmentation can be divided into two steps:

Step 1: As proposed in ConSERT paper, shuffle is the most effective strategies. So we test shuffle with dropout and each cutoff strategies to find the best dropout/cutoff rates.

Step 2: We set the dropout/cutoff rates as the best rates we obtained in **Step 1**. Then test every

[2] https://fanyi.baidu.com/



(a)The original bert.

(b)The ConSERT model.

(c)Model with reverse collapsed.

Figure 8: (a)The bert model with a collapse problem,tends to give a higher similarity.(c) The model with a reverse collapse problem,tends to give sentences the lower similarity score.

	ConSERT-unsup	Translate	ConSERT-sup
Number of train samples	89192	11498	570000

Table 2: Amount of training data.

possible combinations between no augmentation and 5 different augmentation strategies, including shuffle, dropout, token cutoff, feature cutoff and span cutoff. From the tests, we conclude that the combination of shuffle and feature cutoff is the most effective strategy.

4.4 Projection Head

Based on the output of Bert encoder, the projection head we designed map the feature dimension from 768 to hidden layer dimension 600, and consequently map to output dimension 512. According to the analysis from SimCLR (Chen et al., 2020a), by making a comparison between non-linear and linear layes, the setting of projection head with non-linear has a better performance. In our module, we select ReLU as the nonlinear activation and we also use batch normalization to prevent overfitting problem.

Besides, about the setting of projection head using, we mainly refer to SimCSE (Gao et al., 2021), the work consider that keeping MLP during training but removing it at testing time has a better performance than keep it all the time and they eventually select the former. In our work, we keep the same setting with SimCSE.

4.5 Hyperparamter

In our experiment, we find that the temperature hyperparameter has great influence on the results in the comparative learning loss function. It can be seen from the analytical experiment in Table 3 that the optimal result will be obtained when the value is 0.1. This phenomenon once again proves the collapse problem of BERT, because when the sentence representations are all very close, the higher the temperature is the smoother the similarity between sentences will be. And it is difficult for the encoder to learn. At the same time, if temperature is too small, the task is too easy, so it needs to be adjusted to a suitable range.

Temperature	Avg.
0.01	0.663824
0.05	0.7174
0.10	**0.727356**
0.20	0.714259
0.50	0.666623

Table 3: Temperature

Another important hyperparameter is learning rate. The appropriate learning rate can make the objective function converge to the local minimum in the appropriate time. As the Table 10 shows that, learning rate should be in appropriate section.

Learning rate	Avg.
le-7	0.721577
3e-7	0.726973
5e-7	**0.727769**
2e-6	0.727244

Table 4: Learning rate

5 Qualitative Analysis

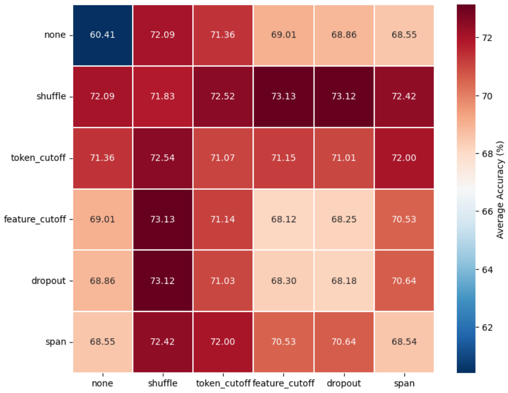


Figure 9: : The performance visualization with different combinations of data augmentation strategies. The row indicates the 1st data augmentation strategy while the column indicates the 2nd data augmentation strategy.

5.1 Sentence Augmentation

As shown in the experiment part, sentence back translation gets the best results. We can observe from Table 8 that, the model with sentence back translation plus shuffle and dropout has the best performance on the test dataset. That's reasonable for the sentence after back translation's meaning will not be changed, and through this process it's distribution will be more similar to the test dataset. Furthermore, the shuffle and the dropout's combination makes the model more related to the downstream STS tasks and improve the model's robustness. For the shuffle directly operated on the token level and change the structure of the sentence and to produce hard examples. And the feature cutoff erases a certain number of features to make the model does correct predictions according to the limited information.

We visualized the results of the sentences after being embedded to validate our results in the experiment part. To visualize the results,we separately extracted the model's embedding layer's output and used the TSNE method to reduct dimensionality of embedding features to a 2-dimensional vector.

From Figure 7, we observe that the back translation's augmentation effect are more similar to the original sentence pairs than the sentence embedding. That's reasonable since the sentence after back translation's meaning will not be changed. But the sentence embedding is not so steady since it will change the sentences from a higher dimensional level.

In addition, we can see from the experiment part, our model performs better than the baseline using unsupervised method. However, our model is in a kind of fuzzy boundary between supervised and unsupervised learning. But contrast with the supervised baseline's 500000 sentences pair size, we approximated it by using only the back-translated STS-benchmark dataset of about 10,000 sentence pairs in total.

5.2 Embedding Augmentation

As described in experiment chapter 4.3, we first test the each dropout/cutoff strategies with shuffle to find the best dropout/cutoff rate, the results are shown in figure 10. The best dropout/cutoff rates are quite different from the rates used in ConSERT. The comparison between ConSERT and ours are shown in table 5.

Next, we consider 6 options for each transformation, including None (i.e. doing nothing), Shuffle, Dropout, Token Cutoff, Feature Cutoff and Span Cutoff, resulting in 6×6 combinations. Note that the Back Translation is not considered here, since it needs additional supervision to generate similar sentences. All these experiments follow the unsupervised setting and use the BERT-base architecture.

The results can be found in Figure 9. We can make the following observations. First, Shuffle and Token Cutoff are the two most effective strategies (where Shuffle is slightly better than Token Cutoff), significantly outperforming Feature Cutoff and Dropout. Second, the performance of Span Cutoff is close to but worse than Token Cutoff, which shows that cutting off continuous tokens is not a good choice. Finally, the best performance is achieved by the combination of Shuffle and Feature Cutoff (0.45 cutoff rate).

5.3 Projection Head

During the design of projection head, we mainly refer to SimSiam (Chen and He, 2021) and SimCLR (Chen et al., 2020a), two contrastive framework of computer vision fields. Simsiam use two MLP structure after encoder, the first MLP serve as projection module and the second is a predictor. One of the representations after projection head would be mapped through predictor and calculate the comparison loss with another representation, it also has a stop-grad strategy to avoid collapsing solutions.

	dropout	feature cutoff	token cutoff	span cutoff
ConSERT	0.2	0.2	0.15	Not test
Ours	0.5	0.45	0.1	0.05

Table 5: Best Dropout/Cutoff Rate Comparison

	MLP	Structure	Avg.
ConSERT	none	none	72.25
ConSERT	simsiam	none	71.48
ConSERT	simclr	single-linear	72.70
ConSERT	simclr	single-noninear	72.73
ConSERT	simclr	multi-hidden512	72.89
ConSERT	simclr	multi-hidden700	72.93
ConSERT	simclr	multi-hidden600	**73.65**
Ours	none	none	75.44
Ours	simclr	multi-hidden600	**76.87**

Table 6: Projection head structure comparison

In our work, we also tried to use this method, but the results achieved were not satisfactory as shown in Table 6.

In contrast, with the addition of the SimCLR-based projection head, both the original ConSERT and our structure with the addition of the sentence augmentation achieve a certain degree of improvement. This proves that the representations after MLP can be better for the calculation of contrastive loss, which ultimately allows the Bert encoder to get better finetuning.

5.4 Early Stop Strategy

To prove the hypothesis that the without early stop mechanism it will cause a reverse collapse problem, we conduct experiments that training models separately with and without the mechanism. The output similarity score and the ground truth value is shown in Figure 8. In general, models which have more points near the diagonal will perform better. We observed that (a) the original Bert with model collapse problem will give the score similarity of the sentence pairs mostly above the 0.7.(b) The Consert model has the best proformance on the similarity calculation but still tends to give the similarity score higher than the groud truth; (c) The model without early stop mechanism, which show a reverse collapse problem, gives lower similarity score than the groud truth.

6 Conclusion

In our work, we first reproduce the experiments under unsupervised setting of ConSERT, and there is a significant improvement over both native Bert as well as Bert-flow on the STS dataset. To further enhance the difference between the two sentence representations in contrastive learning, we add data augmentation before the model embedding layer and inside the Bert model; to better compute the contrastive loss, we add a projection head after Bert encoder; to prevent the contrastive learning model from overtraining, we introduce the early stop mechanism; our approach has a significant improvement over the original ConSERT unsupervised baseline, and achieves a similar performance under the condition that the data level is severely lower than that of the ConSERT supervised baseline.

Acknowledgements

We would like to thank Professor Lingpeng Kong and our TA Yiheng Xu for their dedication to this course and we have acquired a wealth of NLP knowledge. Lastly, we thank Mengzhe He for providing us with many suggestions.

References

Fredrik Carlsson, Magnus Sahlgren, Fredrik Olsson, and Amaru Cuba Gyllensten. 2021. GANDALF: a general character name description dataset for long fiction. In *Proceedings of the 3rd Workshop on Machine Reading for Question Answering*, pages 119–132, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. 2018. Universal sentence encoder for english. In *Proceedings of the 2018 conference on empirical methods in natural language processing: system demonstrations*, pages 169–174.

Pu-Chin Chen, Henry Tsai, Srinadh Bhojanapalli, Hyung Won Chung, Yin-Wen Chang, and Chun-Sung Ferng. 2021. A simple and effective positional encoding for transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2974–2988, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020a. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E Hinton. 2020b. Big self-supervised models are strong semi-supervised learners. *Advances in neural information processing systems*, 33:22243–22255.

Xinlei Chen and Kaiming He. 2021. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15750–15758.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364*.

Ciprian Corneanu, Meysam Madadi, Sergio Escalera, and Aleix Martinez. 2020. Explainable early stopping for action unit recognition. In *2020 15th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2020)*, pages 693–699.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. *arXiv preprint arXiv:1808.09381*.

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

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738.

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

Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*.

Robin Keskisärkkä. 2012. Automatic text simplification via synonym replacement.

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.

Haejun Lee, Drew A. Hudson, Kangwook Lee, and Christopher D. Manning. 2020. SLM: Learning a discourse language representation with sentence unshuffling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1551–1562, Online. Association for Computational Linguistics.

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*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Dinghan Shen, Mingzhi Zheng, Yelong Shen, Yanru Qu, and Weizhu Chen. 2020. A simple but tough-to-beat data augmentation approach for natural language understanding and generation. *arXiv preprint arXiv:2009.13818*.

Yuanmeng Yan, Rumei Li, Sirui Wang, Fuzheng Zhang, Wei Wu, and Weiran Xu. 2021. Consert: A contrastive framework for self-supervised sentence representation transfer. *arXiv preprint arXiv:2105.11741*.

Wangchunshu Zhou, Canwen Xu, Tao Ge, Julian J. McAuley, Ke Xu 0001, and Furu Wei. 2020. Bert loses patience: Fast and robust inference with early exit. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Name	Report	Code	Others
Hanqing Li	Abstract Introduction Methods Experiments Qualitative Analysis Conclusion	Projection Head Sentence embedding visualization	Presentation Model visualization
Chengxiao Li	Methods Experiments Qualitative Analysis	Sentence augmentation Sentence embedding visualization	Code integration Experimental results collation
Yanqiao Li	Methods Experiments Qualitative Analysis	Early stop strategy Sentence embedding visualization	Presentation
Wenbin Zhou	Methods Experiments Qualitative Analysis	Embedding augmentation	Presentation
Chao Fan	Related Work Experiments Qualitative Analysis	Hyperparamter finetuning	

Table 7: Individual Contribution

A Individual Contribution

Our division of labor is shown in the Table 7.

B Best Dropout/Cutoff Rate Selection

In our work, best rate of feature cutoff, token cutoff and span cutoff is 0.45, 0.1 and 0.05. The best rate of dropout is 0.5. The result is shown as Figure 10.

C Experimental results of Back translation with embedding augmentations

The result of sentence augmentation with several embedding augmentations is shown in Table 8.

D Experimental results of hyperparameters finetuning

The results of hyperparameters finetuning are shown in Table 9, Table 10 and Figure 11.

Method	STS12	STS13	STS14	STS15	STS16	STSb	SICK-R	Avg.
ConSERT-unsup	64.70	78.00	68.60	78.91	75.15	73.59	66.77	72.25
translate	68.20	77.86	69.92	79.86	74.77	77.75	69.45	73.96
+shuffle+feature cutoff 0.2	69.61	80.40	71.03	80.20	75.11	76.96	67.71	74.43
+shuffle+feature cutoff 0.45	69.06	80.09	70.46	79.78	75.05	76.83	68.38	74.23
+shuffle+feature cutoff 0.1	69.26	80.31	70.90	80.11	75.17	76.88	68.12	74.39
+shuffle+dropout 0.1	73.09	81.59	72.62	81.04	75.51	77.27	66.95	75.44
+shuffle+dropout 0.2	69.74	80.38	71.00	80.15	75.08	76.95	67.69	74.43
+shuffle+dropout 0.5	68.92	80.00	70.32	79.73	75.04	76.90	68.66	74.22
ConSERT-sup	70.92	79.98	74.88	81.76	76.46	78.99	78.15	77.31

Table 8: Experimental results of Back translation with embedding augmentations

Temperature	STS12	STS13	STS14	STS15	STS16	STSb	SICK-R	Avg.
0.01	0.533373	0.718323	0.607702	0.73840	0.702568	0.671804	0.674597	0.663824
0.03	0.603898	0.748039	0.652451	0.778182	0.727748	0.721367	0.687518	0.702743
0.05	0.630351	0.764029	0.671495	0.786515	0.744919	0.737503	0.686989	0.7174
0.10	0.646412	0.784886	0.690733	0.797203	0.759439	0.739715	0.673106	**0.727356**
0.12	0.648141	0.786623	0.689935	0.795711	0.755411	0.736649	0.667558	0.725718
0.18	0.649729	0.783612	0.68077	0.79048	0.743473	0.723271	0.665266	0.719514
0.20	0.646479	0.779339	0.67505	0.785327	0.737833	0.716998	0.658783	0.714259
0.30	0.624789	0.757024	0.651709	0.764011	0.718166	0.693343	0.638474	0.692502
0.40	0.601123	0.738885	0.635164	0.748135	0.70193	0.678703	0.623985	0.675418
0.50	0.588558	0.726731	0.624597	0.742002	0.693872	0.671937	0.618643	0.666623

Table 9: Temperature

Learning rate	STS12	STS13	STS14	STS15	STS16	STSb	SICK-R	Avg.
le-7	0.645335	0.777411	0.683943	0.787964	0.748836	0.738132	0.6694	0.721577
3e-7	0.646231	0.783592	0.689074	0.795537	0.757923	0.74146	0.674996	0.726973
5e-7	0.64559	0.783519	0.69025	0.797335	0.760355	0.741404	0.675928	**0.727769**
9e-7	0.646719	0.783775	0.689381	0.796196	0.757756	0.741485	0.675109	0.727203
2e-6	0.644515	0.784972	0.690632	0.7964	0.757993	0.741647	0.674611	0.727244
5e-6	0.640687	0.783784	0.692458	0.799011	0.761227	0.740613	0.673821	0.727372
8e-6	0.645895	0.781155	0.686006	0.788532	0.7515587	0.741074	0.6709	0.723589

Table 10: Learning rate

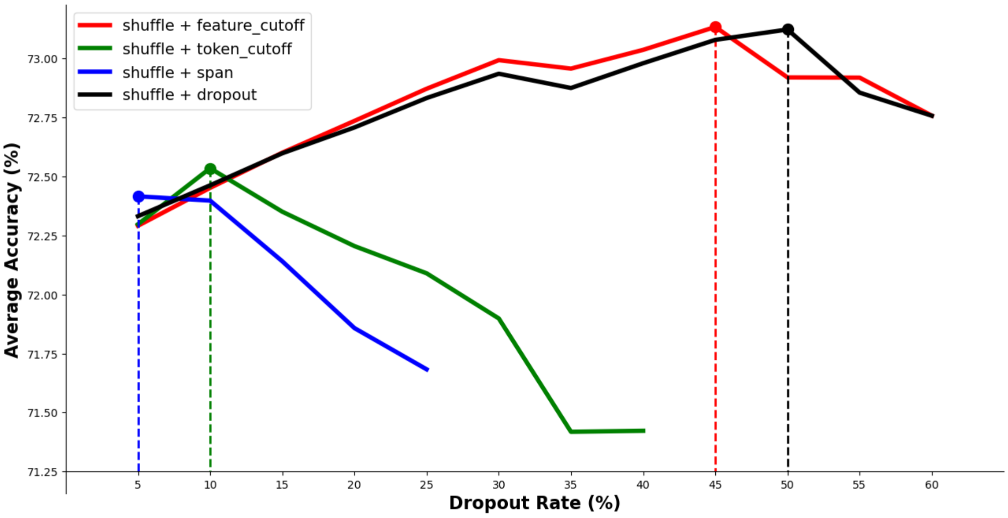


Figure 10: Best Dropout/Cutoff Rate Selection

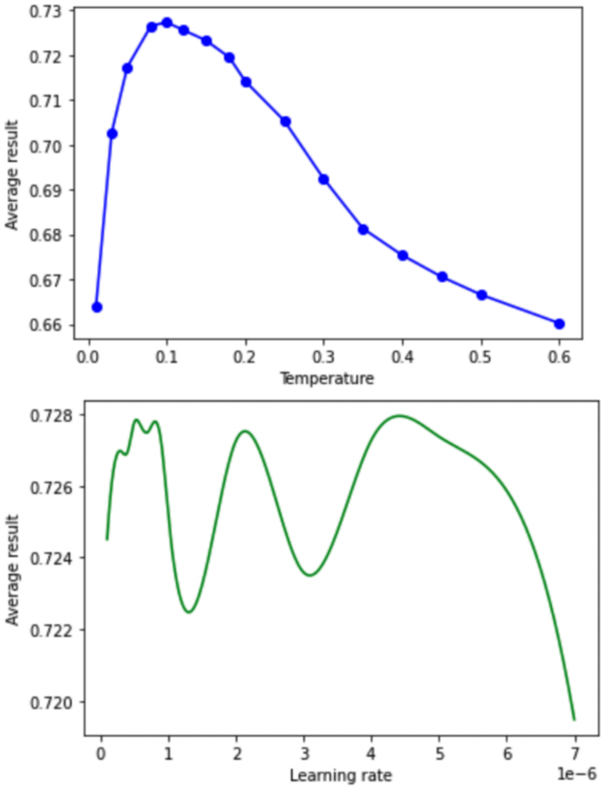


Figure 11: Temperature and learning rate