Combining Contrastive Learning with Adaptive Attention and Experimental Dropout to Improve mini-BERT Performance

Stanford CS224N Default Project

Rachel Liu
Stanford University
rliu25@stanford.edu

Janene Kim
Stanford University
janenek@stanford.edu

Lucy Zimmerman
Stanford University
lucyzim@stanford.edu

Abstract

Contrastive learning has become a valuable tool for handling a variety of downstream NLP tasks. In this project, we aim to identify how the performance of our base minBERT model can be improved by integration of the SimCSE contrastive learning framework and enhance sentence embeddings for downstream tasks. We also introduce cycling iterators with task-weighted losses for balanced training across different datasets. Our findings reveal that SimCSE significantly enhanced sentiment classification by employing dropout as data augmentation to generate robust sentence embeddings. Our best performance was achieved by our final minBERT model with the following extensions: simCSE, dataset-specific weights, task-specific updates, and updated similarity concatenation techniques.

1 Key Information to include

- Mentor: Hamza Al Boudali

2 Introduction

In the rapidly evolving field of NLP, understanding and generating human-like text has remained a significant challenge due to the many subtleties of the human language. This quest has led to the innovation of new transformer architecture and large language models, essential for improving the performance of downstream NLP tasks.

However, traditional state-of-the-art sentence embedding methods have struggled with generating representations that effectively capture the nuanced semantic relationship between sentences, often relying heavily on supervised learning tasks or complex unsupervised methods that do not fully exploit the potential of pre-trained language models (Tianyu Gao, 2021) The focus of our research revolves around the addition of a contrastive learning framework to our base BERT model, and how additionaly adjustments to adaptive attention and dropout may improve performance on downstream tasks.

In this research, we implement a minimally adjusted version of Devlin's BERT large-language model and use pretrained sentence embeddings for three downstream NLP tasks: sentiment analysis, paraphrase detection, and semantic textual similarity (Devlin et al., 2018). We then implement SimCSE, a simple contrastive learning framework proposed by Tianyu Gao (2021), upon the base BERT model, and investigate changes in performance on NLP tasks. Contrastive learning builds superior language representations by pulling semantically close "positive" neighbors close and pushing "negative" sentences apart.

3 Related Work

Within the last decade, natural language processing has been significantly transformed and advanced, with the introduction of language representation models like Devlin et. al's BERT. Devlin et. al present a Bidirectional Encoder Representation from Transformers model that revolutionized the process of pretraining deep bidirectional representations from unlabeled text by conditioning on both the left and right context in all layers (Devlin et al., 2018). This novel approach gave BERT the flexibility to adapt to a wide range of NLP tasks without substantial task-specific architecture modifications, leading to strong and enhanced performance on GLUE scores, SQuAD v.1.1 question answering, and SQuAD v2.0

The promising success of BERT sparked further exploration into enhancing NLP models using novel techniques such as contrastive learning. Logeswaran and Lee (2018) introduced a framework for learning sentence representations from unlabeled data by reformulating the problem of predicting sentence context as a classification task. By employing a form of contrastive learning with a dual-encoder approach where observed contexts were encouraged to be more plausible than contrastive ones, Logeswaran and Lee improved performance on several downstream tasks involving understanding sentence semantics in an order of magnitude faster time (Lajanugen Logeswaran, 2018).

Many authors have attempted to further push modern NLP models via contrastive learning in different contexts. Wieting and Gimple (2015) turned their attention to improving universal paraphrastic sentence embeddings. They explored six different compositional models to encode arbitrary word sequences into vectors with high cosine similarity for sequences with similar meanings, including averaging token vectors and utilizing deep averaging networks (John Wieting, 2015). Later, Gillick et al. (2019) introduced a similar approach in the space of entity retrieval by training a dual encoder model that encodes mentions and entities in the same dense vector space (Dan Gillick, 2010). Thus, candidate entities are retrieved by approximate nearest neighbor search.

Finally, Tianyu Gao (2021)'s SimCSE paper presents a groundbreaking contrastive learning framework for advancing sentence embeddings past state-of-the-art standards. In the unsupervised approach for their model, SimCSE takes an input sentence and predicts itself in a contrastive objective, leveraging standard dropout as noise. Dropout serves as minimal data augmentation, crucial for preventing representation collapse. Although simplistic, this approach performed exceptionally well. The supervised approach incorporates annotated pairs from datasets to encode "entailment pairs" as positives and "contradiction" pairs as hard negatives (Tianyu Gao, 2021). Tianyu Gao (2021)'s SimCSE paper serves as a crucial inspiration for our work, motivating us to explore the potential of contrastive learning and various modifications in enhancing language representation models like minBERT.

4 Approach

4.1 Bert Implementation

As the starting point of our research, we first implemented a minimal BERT model, following the architecture outlined in the original paper by Devlin et al. (2018). The BERT model architecture is composed of a stack of 12 Encoder Transformer layers, as described in the seminal paper by Vaswani et al. (2017).

Specifically, within each transformer layer, we completed the multi-head self-attention component, followed by layer normalizations and a position-wise feed-forward layer, and finally applied dropout with a setting of $p_{drop} = 0.1$.

4.2 Downstream Tasks

For each downstream task, we incorporated task-specific layers atop BERT's output, enabling targeted predictions tailored to the nuances of each task. In particular, linear transformations are applied, which allow the model to learn complex patterns and representations, and also output logits. We also employed task-specific loss functions, including cross-entropy loss for sentiment classification, binary cross-entropy loss for paraphrase detection, and the SimSCE loss function for semantic similarity. These tailored loss functions facilitated effective optimization for each task.

To address the task of sentiment classification, we appended a linear layer atop BERT's output to predict sentiment classes. We utilized cross-entropy loss for training and applied dropout regularization to prevent overfitting.

In paraphrase detection, we engineered concatenated features to capture semantic nuances between sentence pairs, leveraging the absolute difference and sum of embeddings, as seen in John Wieting (2015). This feature engineering technique enhanced the model's ability to discern paraphrases accurately. We employed binary cross-entropy loss for training, as there was only one output logit, with logits passed through a sigmoid function for prediction.

For sentiment classification, in addition to the linear transformation layers we applied, we also utilized ReLU activation layers, which introduces non-linearity to the model, enabling it to learn more intricate decision boundaries, which can be crucial for tasks like sentiment classification where the relationships between input features and sentiment labels may not be linear. Vinod Nair (2010) and then calculated the cross-entropy loss. This was summed with our SimSCE loss function, used contrastive learning and cosine similarity between sentence pairs is computed to assess their semantic similarity.

4.3 SimCSE Integration

Upon our base BERT model, we implemented the SimCSE contrastive learning framework as proposed by Tianyu Gao (2021). This integration aims to leverage unsupervised learning to generate more discriminative sentence embeddings.
In the SimCSE framework, we create positive pairs by passing the same sentence through the BERT model twice, applying standard dropout, and thus obtaining two different 'positive' embeddings. Other sentences in the same mini-batch are treated as negative pairings.
Given a sentence s, we generate two embeddings, h, and h', by applying the BERT model with independently dropout masks:

$$h = BERT_{dropout1}(s) h' = BERT_{dropout2}(s) \tag{1}$$

Contrastive learning aims to maximize the similarity between positive pair h and h' while minimizing the similarity between h and embeddings of other sentences in the batch (**??**). The loss function for a single positive pair h, h' in a mini-batch of size N is defined as:

$$\mathcal{L} = -\log \frac{e^{\frac{sim(h,h')}{\tau}}}{\sum_{i=1}^{N} e^{\frac{sim(h,h_i)}{\tau}}} \tag{2}$$

Where $sim(h, h')$ is the cosine similarity calculated between embeddings h and h', and τ is a temperature hyperparameter that scales the similarity scores. z is the standard dropout mask in the original BERT model, and was not modified. The value, 0.1, for the probability of the dropout layer was calculated by Tianyu after rigorous experimentation and optimization.

4.4 Cycling and stabilization

We employ cycling iterators, as opposed to padding or truncating data, to ensure balanced exposure to data from each task during training. Task-weighted losses are utilized to address dataset imbalances and optimize task-specific performance. These losses are inversely proportional to each dataset's size to address any potential bias stemming from dataset size discrepancies. By assigning higher weights to smaller datasets, our model is incentivized to allocate more resources towards learning from these datasets, thereby mitigating the impact of dataset imbalances on model performance. Additionally, the utilization of cycling iterators ensures that each task receives equitable attention during training, minimizing the risk of overfitting to a particular task or dataset. Gradient clipping is applied to further stabilize training.

4.5 Optimizer

Furthermore, we employed the Adam optimizer, a stochastic optimization method that adjusts learning rates for various parameters. Our implementation includes the step() function, which updates

exponential moving averages of gradients and squared gradients, performs bias correction, and incorporates weight decay regularization, referencing Loshchilov and Hutter (2017). We adopted Kingma and Ba's Adam optimization algorithm as our baseline (Kingma and Ba, 2014).

5 Experiments

5.1 Data

Our base minBERT model was pre-trained using two unsupervised tasks – masked token prediction and next sentence prediction – on Wikipedia articles. We utilized distinct datasets across three different downstream NLP tasks to fine-tune and evaluate the performance of minBERT model. Below, we detail the datasets we employed, including their structure, content, and the tasks they facilitate.

For sentiment analysis, we leverage two datasets: the Stanford Sentiment Treebank (SST) dataset and the CFIMDB dataset. The SST dataset comprises of 11,855 sentences from movie reviews, extracted and parsed to include a total of 215,154 unique phrases. Each phrase is annotated with one of five sentiment labels: negative, somewhat negative, neutral, somewhat positive, and positive(Socher et al., 2013). This granularity allows for a nuanced understanding of sentiment beyond simple binary classification. The CFIMBD dataset contains 2,434 highly polar movie reviews labeled as positive or negative, with a focus on distinguishing between extreme sentiments (Maas et al., 2011).

For the task of paraphrase detection, we used the Quora Dataset, which consists of 400,000 question answer pairs. Each pair is annotated with a binary label indicating whether the pair is a paraphrase of each other. Lastly, to assess semantic textual similarity, we used the SemEval STS Benchmark dataset that features 8,628 sentence pairs. Each pair is scored from 0 (indicating no relation) to 5 (indicating equivalent meaning).

Each dataset is fine-tuned separately on the pre-trained minBERT model to adapt its embeddings to a specific task. We loaded pre-trained model weights into minBERT and adjusted these embeddings through task-specific training on each dataset.

5.2 Evaluation method

For our evaluation, we utilize the evaluation metrics outlined in the CS 224N default project handout. In order to assess the performance of our base minBERT implementation without extension, we compare our accuracies against mean reference accuracies for the SST and CFIMDB datasets. For multitask tasks, we measure our model quality by accuracy on SST and QQP and Pearson correlation on STS. We also utilize overall dev score, which is calculated by normalizing the score of each downstream task between 0 and 1, and then averaging those three metrics together.

5.3 Experimental details

Base learning rate and number of epochs BERT layers include a dropout rate of 0.1, while the classification head employs a dropout rate of 0.3. In each experiment, we ran 10 epochs with a fine-tune learning rate of 1e-5 and pretrain learning rate of 1e-3. This was deemed sufficient to achieve maximum performance on the development set, as we observed peak performance on the development set is observed around the 3rd to 4th epoch. Thus, unless stated otherwise, the learning rate of 1e-5 is kept constant for further experiments. The best development set performance over 10 epochs on training sets is reported.

Non-linearity for Sentiment Classification For sentiment classification, we examine the idea of introducing non-linearity and additional linear transformation layers to improve accuracy. It was found that introducing non-linearity to the model, in the form of additional ReLU activation layers, as from (Vinod Nair (2010)), enhanced the results of the mean accuracy for the SST database. Three linear layers (sentiment linear, sentiment linear1, sentiment linear2) are applied sequentially to transform the BERT embeddings. Each linear layer has an input size of config.hiddensize and an output size of config.hiddensize. This configuration allows for non-linear transformations of the BERT embeddings. This was found to be effective in bolstering our SST accuracy. We discovered the numbers were better reflected once the datasets were weighted.

Table 1: Comparison of Baseline and Our minBERT Implementation Results

Multiple linear and non-linear layers (weighted)	One Linear Layer	Multiple linear and non-linear layers (unweighted)
Pretraining for SST	0.390 (0.007)	0.409
Pretraining for CFIMDB	0.780 (0.002)	0.788
Finetuning for SST	0.515 (0.004)	0.530
Finetuning for CFIMDB	0.966 (0.007)	0.971

Table 2: Comparison of Linear and non-linear layers

	One Linear	Mult. Unweighted	Mult. Weighted
SST Accuracy	0.316	0.319	0.345

Paraphrase Concatenation For paraphrase detection, we examine the idea of concatenating the absolute difference and sum between embeddings, as from John Wieting (2015) We experimented at first with concatenating the two embeddings. Then it was discovered that concatenating the two values for absolute difference and absolute sum improved the mean accuracy for paraphrase detection.

Table 3: Comparison of Paraphrase Embedding Concatenation

	Concatenating Embeddings	**Concatenating the Absolute Difference and Sum**
Paraphrase Accuracy	0.379	0.455

Cycling To incorporate data from all three datasets, it was necessary to experiment an approach to iterate through data. We experimented with truncating all the datasets to the size of the smallest dataset, STS. This was found to be the least effective method to iterate through data based on the total manually calculated mean. Then we experimented with padding the smaller datasets with 0s, which then flagged those values to not contribute that respective loss. This improved our overall mean accuracy, but significantly improved the paraphrase detection accuracy. Finally, we implemented cycling iterators that ensured that each task is sampled equally over multiple iterations. This further improved the mean.

Table 4: Comparison of different iteration methods

Method	**STS Accuracy**	**SST Accuracy**	**Paraphrase Accuracy**	**Mean**
Truncating	-0.020	0.314	0.380	0.225
Padding	-0.014	0.315	0.420	0.240
Cycling	-0.002	0.319	0.455	0.257

Weights and Stabilizations We experimented with weighting the respective calculated loss for each dataset. This immediately equalized the improvement across datasets, as opposed to seeing a concentrated improvement with just the SST. We experimented with an equal spread weighting each dataset equally. Then, we experimented with weighting the data inverse proportionally to the size of the dataset, which was found to be the most impactful method. Gradient clipping was also found to enhance the accuracy.

Table 5: Comparison of Weights

Method	**STS Acc.**	**SST Acc.**	**Paraphrase Acc.**	**Overall Dev**
Equal Weights	-0.002	0.319	0.455	0.424
Proportionally Inverse + Gradient clipping	0.079	0.345	0.513	0.466

SimCSE To further enhance our model, we experimented with contrastive learning, based on the SimCSE contrastive learning framework as proposed by **?**. We experimented with applying different drop-out to a preliminary embedding to generate a second embedding. We found the 0.1 value to be most successful in maximizing accuracy. In addition, we attempted to implement SimSCE for other downstream tasks including paraphrase detection, but found little improvement. As a result, we only

computed SimSCE for sentiment classification. In addition, we experimented with considering this contrastive loss as our sole loss calculation for sentiment. Then we experimented with summing this contrastive loss with the cross-entropy loss calculated already for sentiment classification. This was where we saw results improve.

Figure 1: Comparison of SST Accuracy using Different Dropout Variations

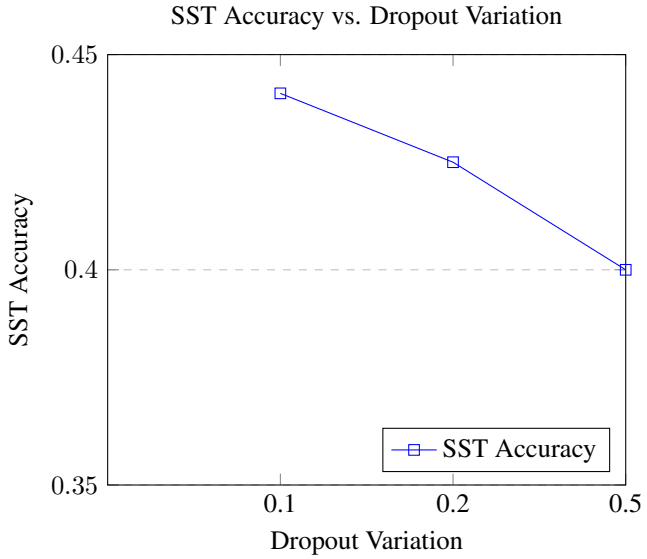

Table 6: Comparison of SST Accuracy using Different Loss Calculation Methods

Loss Calculation	**Only contrastive_loss**	**Contrastive_loss + Cross_entropy_loss**
SST Accuracy	0.402	0.441

5.4 Results

In order to evaluate all our different iterative models, the evaluation metrics we utilized were accuracy for the sentiment analysis and paraphrase detection tasks, and Pearson Correlation for the semantic textual similarity task.

Table 7: Comparison of Baseline and Our minBERT Implementation Results

Task	**Baseline Dev Accuracy**	**Our Dev Accuracy**
Pretraining for SST	0.390 (0.007)	0.409
Pretraining for CFIMDB	0.780 (0.002)	0.788
Finetuning for SST	0.515 (0.004)	0.530
Finetuning for CFIMDB	0.966 (0.007)	0.971

In Table 7, we report the dev set performance of our base minBERT implementation for sentiment classification on the SST and CFIMDB test sets as compared to the benchmarks from the default project handout.Our results closely align with the project handout baseline dev accuracies, which confirms to us that our base minBERT implementation is correct. In fact, our results are slightly higher, but since it is not by a significant margin, we have confidence that we implemented minBERT in just the standard way.

In Table 8, we report the dev performance for all of our proposed iterative models on the SST, Paraphrase, and STS downstream tasks. We evaluate various multitask learning approaches that combined our minBERT with different extensions like updated similarity concatenation, a downstream task update (includes paraphrase concatenation and non-linearity sentiment classification), gradient clipping, weights for each dataset, and of course our main extension of implementating simCSE.

Table 8: Model Performance Improvement with Iterative Extensions

Attempt	SST dev accuracy	Paraphrase dev accuracy	STS dev correlation	Overall dev score
1st attempt (Base minBERT) (DEV)	0.321 (+0.321)	0.379 (+0.379)	-0.059 (+0.941)	0.390 (+0.390)
2nd attempt + updated similarity concatenation (DEV)	0.316 (-0.005)	0.379 (+0.000)	-0.034 (+0.025)	0.393 (+0.003)
3rd attempt + downstream task update (DEV) *	0.319 (+0.003)	0.455 (+0.076)	-0.002 (+0.032)	0.424 (+0.031)
FINAL attempt + grad. clip. + weights + simCSE (TEST)	**0.441 (+0.124)**	**0.675 (+0.297)**	**0.269 (+0.304)**	**0.583 (+0.191)**

Note: "downstream task update" includes paraphrase concatenation and non-linearity sentiment classification implemented so that we train on all 3 of the tasks. The table shows the performance improvement of each subsequent attempt, where each attempt includes the extensions listed.

The quantitative results illustrate a progressive improvement across our several evaluation metrics. The improvement from the first to final attempt was significant, and was better than expected. Notably, the final attempt, which incorporated weights for theLM dataset and SimCSE, showed significant gains in paraphrase development accuracy and STS development correlation, which also was surprising. The overall development score also saw a considerable increase, indicating that the integrated modifications contributed positively to the model's ability to perform across multiple tasks.

These outcomes suggest that the contrastive learning framework and task-specific optimizations are highly effective in enhancing the model's performance. The application of contrastive learning, in particular, seems to have provided a robust method to generate discriminative sentence embeddings, a critical component for tasks like sentiment analysis. The step-wise enhancements and the detailed exploration of various techniques provide a comprehensive roadmap for improving a model's performance on NLP tasks. These results affirm the effectiveness of iterative development, and the importance of model-specific adaptations.

6 Analysis

In terms of the model itself, The minBERT model, a scaled-down version of the BERT architecture, serves as the foundation. Given that BERT's architecture is highly capable but also resource-intensive, minBERT likely aims to balance performance with computational efficiency. The initial performance may not have been groundbreaking due to its reduced complexity; however, it established a baseline to gauge the impact of subsequent enhancements.

The limited accuracy in the early iterations may not have introduced sufficient non-linearity. Advanced NLP tasks often require complex transformations that simple linear models cannot capture. Non-linearity, introduced through activations like ReLU (Vinod Nair, 2010), was shown as essential for modeling the complicated relationships in language. In addition, the similarity measures used in early iterations were not sophisticated enough, and the model might not have been capturing the depth of semantic relationships necessary for tasks like STS.

The second attempt added updated similarity concatenation techniques, which initially did not lead to substantial improvements. This method, drawing from techniques such as those described by John Wieting (2015) for learning structured text representations, enhanced the model's ability to capture the semantic nuances between sentences. However, the limited improvement suggests that while the model could better understand relationships between sentence pairs, it still lacked a comprehensive understanding of sentence embeddings in isolation, which is crucial for tasks like STS where individual sentence representations are compared.

The introduction of task-specific updates, which included fine-tuning BERT's output layers with additional transformations and applying task-oriented loss functions, were crucial for optimizing the model's predictions for each specific task. These allowed the model to fine-tune for the idiosyncrasies of each task, which is essential for specialized tasks. The third attempt included task-specific updates such as linear transformations and specialized loss functions. For instance, for sentiment classification, the addition of ReLU activation layers introduces non-linearity that helps in modeling complex patterns. However, the improvements were marginal. This could indicate that while the model began to adapt better to the nuances of specific tasks, the overall representational power of the sentence embeddings was not significantly enhanced.This initial implementation needed fine tuning, adjusting specific layers' methods such as within similarity and paraphrase, suggesting that while the

model was being better tailored to the tasks, the underlying representations may still have required refinement.

The final attempt incorporated dataset-specific weights and SimCSE, a contrastive learning framework. By weighting the loss for each dataset, the model addressed the imbalance in data representation. This implies that previously, the model might have been biased towards the larger datasets, and smaller datasets were underrepresented in the loss calculation. The model's improved performance after adjusting weights suggests that each dataset now contributed more evenly to the model's learning process, leading to better generalization across tasks.

Significant improvements for sentiment classification came from integrating SimCSE, which uses dropout as a form of noise to create positive and negative sentence pairs for contrastive learning. The improved performance suggests that dropout as noise allowed the model to generate more diverse and robust sentence embeddings. The model learned to pull together embeddings of the same sentence (positives) and push apart different sentences (negatives), refining the sentence space such that semantically similar sentences are closer. This success is partly because contrastive learning effectively encourages the model to learn distinct representations for different sentences. The unsupervised nature and the novel use of dropout as data augmentation allowed the model to better differentiate between sentences, enhancing its understanding of sentence semantics without the need for extensive labeled data.

Additionally, using cycling iterators and gradient clipping provided balanced exposure and stable training across tasks, preventing overfitting and underrepresentation of any single task.

The iterative enhancement process reveals the nuanced nature of NLP model optimization. It demonstrates the significance of balancing task-specific knowledge with general linguistic understanding. By examining these extensions and their outcomes, we see the interplay between model architecture, training techniques, and the importance of a balanced dataset representation in developing a robust NLP system.

7 Conclusion

Our project revolved around the adaptation and enhancement of Devlin's BERT model and implementation of the SimCSE contrastive learning framework in order to refine sentence embeddings.

Our main findings revealed that compared to the baseline BERT model, our model with SimCSE integration and adjustments for task-specific optimizations significantly elevated performance on downstream tasks. Furthermore, the iterative introduction of additional non-linearity through ReLU activation layers and the adjustment of task-specific layers and loss functions led to a nuanced model capable of discriminating fine semantic differences.

However, our work primarily leveraged unsupervised tasks due to dataset limitations, potentially overlooking the benefits of supervised fine-tuning on a broader range of datasets. Looking ahead, future research could including applying our methods to a wider array of NLP tasks and experimenting with supervised fine-tuning techniques on diverse datasets. Moreover, investigating the potential of novel data augmentation methods and the deeper integration of multitask learning could continue to push the boundaries of NLP.

References

Larry Lansing Alessandro Presta Jason Baldridge Eugene Ie Diego Garcia-Olano Dan Gillick, Sayali Kulkarni. 2010. Learning dense representations for entity retrieval. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 528–537, Online. ACL Anthology.

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*.

Kevin Gimpel Karen Livescu John Wieting, Mohit Bansal. 2015. Towards universal paraphrastic sentence embeddings. In *International Conference on Learning Representations*, page 0, Online. arXiv.

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

Honglak Lee Lajanugen Logeswaran. 2018. Simcse: An efficient framework for learning sentence representations. in international conference on learning representations. In *International Conference on Learning Representations*, page 0, Online. arXiv.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

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

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Danqi Chen Tianyu Gao, Xingcheng Yao. 2021. Simcse: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762*.

Geoffrey E. Hinton Vinod Nair. 2010. Rectified linear units improve restricted boltzmann machines. In *ICML 2010*.