# Training MinBERT with Contrastive Learning

Stanford CS224N Default Project

**Robert Thompson**
Department of Computer Science
Stanford University
rthomp@stanford.edu

**Salvatore Spina**
Department of Computer Science
Stanford University
salspina@stanford.edu

**Patrick Donohue**
Department of Computer Science
Stanford University
pdonohue@stanford.edu

## Abstract

BERT models are extremely powerful and generalizable tools for a variety of downstream NLP tasks. However BERT models often suffer from anisotropic sentence embeddings, reducing the ability of the model to differentiate sentences on qualities like sentiment and semantic similarity. We applied contrastive learning techniques to minBERT to reduce anisotropy. This along with several other model techniques including gradient surgery, task interleaving, and hyperparameter search allowed us to significantly improve minBERT on three downstream tasks: paraphrase detection, sentiment analysis, and semantic textual similarity. When comparing against a baseline bert model, our accuracy on the first two tasks improved from .38 to .71 and .31 to .49, respectively. Our Pearson correlation for STS improved from -.10 to .35 on the final task.

*We would like to thank our mentor, David Huang.*

## 1 Introduction

Large Language models have set a new standard for nearly all NLP tasks. However, these large pretrained models on their own do not always generate optimal embeddings for every task. Prior work has shown that sentence embeddings from pre-trained language modelings can be limited by the learned anisotropic word embedding space. Because of this, these models sometimes have difficulty generalizing to downstream tasks. Since the emergence of large language models, there has been lots of work surrounding how to further pretrain and finetune these embeddings to produce better results. SimCSE, proposed by in *SimCSE: Simple Contrastive Learning of Sentence Embeddings*[1], implements contrastive learning to Bert base in order to achieve higher scores on Semantic Textual Similarity scores. Inspired by this approach, we first wanted to see how much of an effect would implementing contrastive learning on minBERT have. Although it is a simpler model than BERT, would this novel approach still be able to deliver superior embeddings?

To build on this paper, we wanted to see how well contrastive learning techniques could do on generalizing to downstream tasks. Gao *et al*did not attempt this. Specifically, we wanted to use contrastive learning to generate embeddings that apply to textual similarity, sentiment prediction, and paraphrase prediction.

First, we determined that finetuning using the simCSE loss function on the NLI dataset[2] improves performance.

Next, we performed a hyperparameter search on $\tau$ (the simCSE loss function temperature parameter), along with other parameters including learning rate and dropout probability. We found best results when setting $\tau$ significantly higher than Gao *et al*did in their paper.

We pretrained BERT embeddings and layers using just the simCSE supervised task. We then twice finetuned just the final linear layers for each of our three downstream tasks, once on the original BERT layers and once on our simCSE-finetuned layers. We found improved results when BERT embeddings and layers had been pretrained with simCSE (see Figure 2).

Finally, we applied gradient surgery, projecting the gradients from each of our four training tasks onto one another. We ran experiments, determining that gradient surgery only helps for a relatively high learning rate.

## 2 Related Work

Wang and Isola (2020) [3] proposed two measurements for the quality of embedding: alignment and uniformity. In a well aligned set of embeddings, semantically-related pairs occupy nearby points in vector space. This allows a model to draw relationships. In a uniform set of embeddings, if each embedding is normalized then the set as a whole is evenly (or uniformly) spread across the unit sphere. This maximizes the amount of information that can be encoded in embeddings.

Recent work by Ethayarajh (2019)[4] identifies an anisotropy problem in language representations. Instead of being evenly distributed on the unit sphere, embeddings occupy a narrow cone in the vector space. Because of this, similarity measurements like cosine similarity are not very effective. As a result, some much simpler models like GloVe actually score higher than models like BERT on semantic textual similarity (STS) tasks. This limited occupancy of space in the vector space is called anisotropy. Ethayarajh (2019) showed this can occur in pre-trained contextual representations. Wang and Isola (2020) [3] show how anisotropy is connected to uniformity. Intuitively, optimizing the contrastive learning objective can improve uniformity (or ease the anisotropy problem), as the objective pushes negative instances apart.

Gao *et al*[1] demonstrated how a contrastive learning objective leads to more uniform and better-aligned embeddings. As shown in figure 1, constrastive learning takes an anchor sentence, a positive example, and a negative example, and attempts to move positive pairs closer together and negative ones farther. The authors conclude this leads to greater alignment and uniformity. Contrastive learning can alter the latent embedding space, which leads to improved performance on STS tasks.

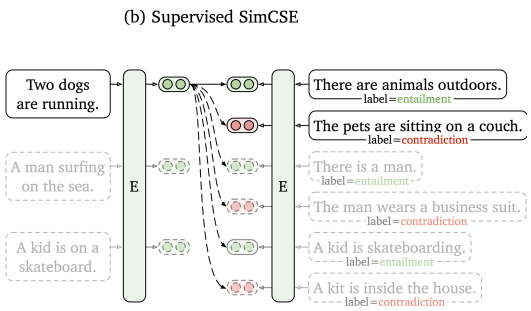


Figure 1: Supervised SimCLE

## 3 Approach

For our baseline model we implemented vanilla minBERT as described in the BERT paper [5]. At a high level, BERT is a bidirectional model, meaning it is trained to look both forwards and backwards in a text sequence to capture context. It uses a masked language modeling (MLM) task and a next sentence prediction (NSP) task to train the model. In the MLM task, BERT is trained to predict missing words in a sentence by masking some of the words and requiring the model to fill in the blanks. In the NSP task, BERT is trained to predict whether two sentences follow each other in the original text or not.

After implementing our baseline, we followed a similar approach to [1], and used our pretrained embeddings to train over an NLI dataset, using the following contrastive loss equation that they defined:

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

$\tau$ is a temperature hyperparameter, which we changed experimentally (see Experiments section). *sim* is short for cosine similarity. Note that while Gao *et al*has published their own code, we implemented our own version as to make it work in conjunction with our model.

Each row in the NLI dataset contains a given sentence premise $x_i$, an entailment $x_i^+$, and a contradiction hypothesis $x_i^-$. We use our pretrained embeddings to generate $h_i$, $h_i^+$, $h_i^-$ respectively. We used cosine similarity function.

Like the original paper, we then evaluated our contrastive model over an STS dataset.

We sought to expand this paper by implementing multitask training in order to see if our model could generalize well across multiple tasks. We first used round-robin multitask learning to train across the 4 datasets and tasks described in our data section. Our model alternates at each gradient step between updating the parameters for each task during training. At each iteration, the network processes a mini-batch of data from one of the tasks and updates the network parameters based on the loss function for that task. It then moves on to the next task in the cycle and repeats the process. This continues until all tasks have been processed for one epoch.

After this, we hypothesized that our gradients across the different tasks may be competing with each other, so we implemented PCGrad gradient surgery to see if this could improve our multitask learning. To implement this we used a pre-implemented Pytorch library PCGrad. [6].

PCGrad gradient surgery works by projecting the gradients onto a lower-dimensional space. Specifically, it computes the gradient direction with respect to the parameters and the gradient direction orthogonal to the parameters as specified in equation 2. It then scales the gradient direction with respect to the parameters to a maximum norm and keeps the gradient direction orthogonal to the parameters unchanged. This approach effectively reduces the magnitude of the gradient while preserving the direction of the gradient. If our gradients are in conflict directions, this will lead to more stable training and faster convergence.

$$g_i = g_i - \frac{g_i \cdot g_j}{||g_j||^2} \cdot g_j \tag{2}$$

# 4 Experiments

## 4.1 Data

**Stanford Sentiment Treebank (SST)** As described in the project handout "the Stanford Sentiment Treebank consists of 11,855 single sentences extracted from movie reviews. The dataset was parsed with the Stanford parser2 and includes a total of 215,154 unique phrases from those parse trees, each annotated by 3 human judges. Each phrase has a label of negative, somewhat negative, neutral, somewhat positive, or positive." [7]. This dataset is used for training and evaluating sentiment classification.

**Quora** As described in the project handout "The Quora dataset [...] consists of 400,000 question pairs with labels indicating whether particular instances are paraphrases of one another." We used the following splits.
• train (141,506 examples)
• dev (20,215 examples)
• test (40,431 examples) [7]
This dataset is used for paraphrase prediction.

**SemEval STS Benchmark** As described in the project handout "The SemEval STS Benchmark dataset [...] consists of 8,628 different sentence pairs of varying similarity on a scale from 0 (unrelated) to 5 (equivalent meaning)." For the STS dataset, we used the following splits:
• train (6,041 examples)
• dev (864 examples)
• test (1,726 examples) [7].
This dataset is used to predict semantic textual similarity.

**NLI** From Gao *et al*, we use their dataset. Specifically, they combined SNLI dataset(Bowman et al., 2015)[8] and MNLI (Williams et al., 2018)[9]

Each row contains a given sentence premise, an entailment, and a contradiction hypothesis. A row from the data could look like ("uh you know it 's if some if a teacher does anything that uh they 're liable to have a law suit against them for uh cruel and unjust punishment or whatever", "if a teacher does something untoward, legal action might be taken against them for cruel/unjust punishment .", " no legal action can be taken against a teacher for cruel/unjust punishment.") There are 310,000 triplets in our dataset, and we trained on 70% of them. We did no evaluation on this dataset.

### 4.2 Evaluation method

On the Quora paraphrase dataset dataset we evaluate ourselves based on accuracy percentage. This is a binary prediction task.

For sentiment classification on the SST dataset we again use accuracy percentage, however this time we have five classes.

For STS, we calculate the Pearson correlation of the true similarity values against the predicted similarity values across the test dataset. Specifically:

$$\rho_{X,Y} = \frac{\text{cov}(X,Y)}{\sigma_X \sigma_Y} = \frac{\sum_{i=1}^{n}(X_i - \overline{X})(Y_i - \overline{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \overline{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \overline{Y})^2}} \tag{3}$$

Throughout our paper we primarily report a model's 'Total Score' or 'Average Score,' which is the average on the three metrics discussed above.

### 4.3 Experimental details

During our training process, we conducted hyperparameter searches for Learning Rate, Dropout probability, and number of Epochs. We also used played around with training on all tasks, just the NLI contrastive learning task, and all tasks except the NLI task. Full results of these searches are in the appendix.

Contrastive learning is designed to improve the latent space of embeddings, setting a language model up for improved performance on all tasks. Thus we pretrained all of our layers on the NLI task, and then did a quick finetune of just the linear classifier heads of our model. It turns out that training on the NLI task improves performance for all three of our downstream tasks, not just the STS task it most resembles. In fact, we see greatest improvement in the paraphrase detection task. This suggests that training using the contrastive learning loss function does improve the uniformity of our embedding space. See Figure 2.

We performed a hyperparameter search on $\tau$, the temperature parameter from Gao *et al*'s paper. They suggested using $\tau = 0.05$. However after using this setting for several small tests (n_epochs = 3), we found that much higher values of $\tau$ performed better. Below is a graph showing a hyperparameter search for a much larger number of epochs - as the training time increases, smaller values of $tau$ begin to improve relatively. Therefore it is likely that Gao *et al* found a different optimal value of $tau$ because they trained bigger models for much longer. See figure 3.

After adding gradient surgery, we performed an experiment to determine whether using it would improve performance. For low learning rates, we found that using gradient surgery did not impact or diminished performance. For a lr that is 3 times our usual, however, we found that using the pcgrad library did improve performance. This makes intuitive sense: gradient surgery necessarily shrinks

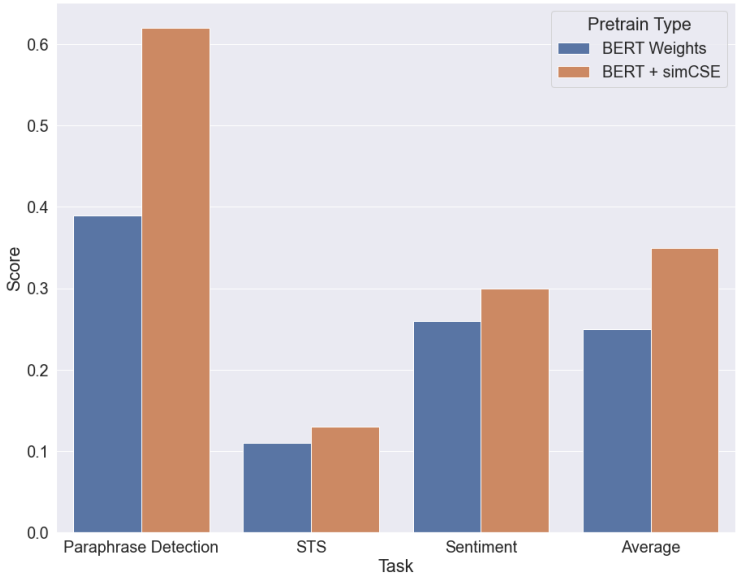


Figure 2: Pretrained Performance: MinBERT vs MinBERT plus simCSE

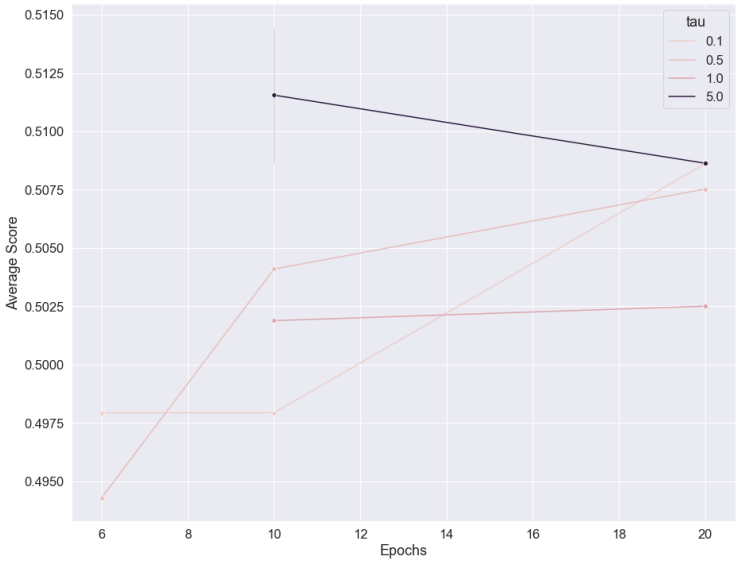


Figure 3: Tau Hyperparameter Search

gradients by projecting them, so to counteract this learning rate should be increased. See figure 4, where we performed a search while training for 8 epochs).

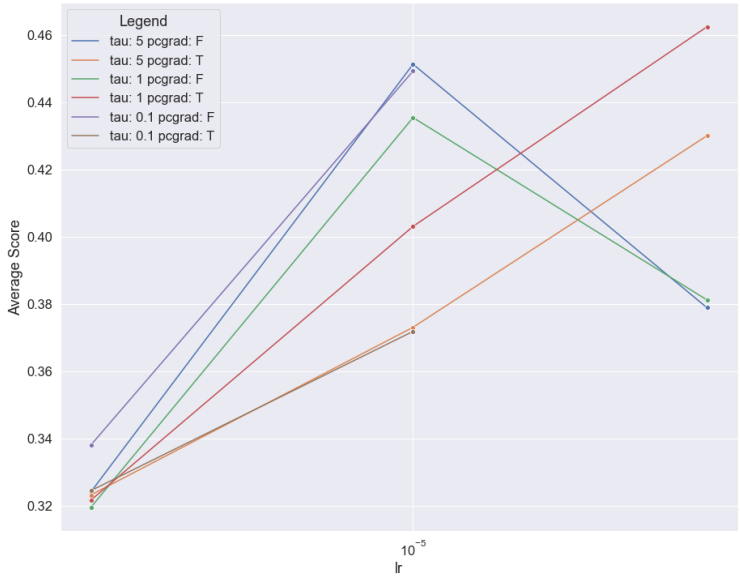


Figure 4: Does Gradient Surgery Improve Performance?

The final row of table 1 in the appendix indicates the hyperparameter values on our best results on the dev set.

### 4.4 Results

We expected to see improvements after implementing constrastive learning namely because of increased alignment and uniformity. This did in fact turn out to be the case. See figure 2.

We expected gradient surgery to improve performance consistently, but it did not. We would have loved to try more experiments with this method and even higher learning rates, but were limited by time and GPU compute.

## 5 Analysis

We looked at actual inputs to our various three tasks, and analyzed why they failed. Below we go through some examples:

**Paraphrase Detection** (Quora Dataset):

Our model struggled with rarely seen works and tokens, which is understandable considering that is small relative to modern LLMs. For example, our model predicted incorrectly that that these two sentences were paraphrases of one another: *"How can I learn megruli?"* and *"How does one learn to learn?"* This is likely the only time that the model sees the word "megruli" in training (this example is pulled from the training set).

Sometimes it is also genuinely ambiguous whether or not two sentences are paraphrases. These two are marked as paraphrases: *"I have lost my Aadhaar enrollment slip and registered mobile number as well. How do I download my Aadhaar card?"* and *"I don't have an enrollment number or an Aadhaar card number. What should I do?"* As far as Quora is concerned, these two people likely need the same advice. But one person lost their card number and enrollment slip, whereas the other person never had one in the first place, so one could argue that they aren't true paraphrases. Our model just tries to predict whether people convey the same meaning, and not whether their goals are the same.

**Semantic Textual Similarity** (STS Dataset):

On the STS dataset, our model failed when it could not determine which words in a sentence were unimportant. For example, for the sentence pair *"Microsoft acquired Virtual PC when it bought the assets of Connectix earlier this year"* and *"Microsoft acquired Virtual PC from its developer, Connectix, earlier this year"* our model gave an STS score of 2.8/5, whereas the dataset had 4.4/5. "Bought the assets" and "from its developer" convey little or no additional meaning in this sentence - they are fluff. But our model can't tell because it has a relatively small model of the world, and so the STS score is lower than it should be. A similar example is the following: *"Hundreds of Bangladesh clothes factory workers ill"* and *"Hundreds fall sick in Bangladesh factory,"* which give an STS score of 2.1/5 when the actual score is 4.4/5. Likely the model does not realize that the word "clothes" is not important to this sentence. Based on other examples we believe that it recognizes the similarity of "fall sick" and "ill."

**Sentiment Analysis** (SST Datset):

The sentiment task was evaluated on just accuracy percentage and not degree of closeness. There can often be conflicts in the 1-5 scale rating system, which further highlights the importance of how our data is created and labeled. For instance, the sentence *"Post 911 the philosophical message of Personal Freedom First might not be as palatable as intended"* was ranked 1 out of 5 by our model. Our team independently also ranked some sample sentences and gave this a 1, however our dataset gives the sentences a 2. Despite the close-ness, our model's performance score was impacted for this arguable discrepancy.

These sorts of sentiment discrepancies can happen on the positive end of the spectrum as well. For instance, the example *"The gorgeously elaborate continuation of " The Lord of the Rings " trilogy is so huge that a column of words can not adequately describe co-writerdirector Peter Jackson's expanded vision of J.R.R. Tolkien 's Middle-earth"* was ranked a 3 by our model, which seems plausible. However the data has the sentence ranked a 4. Qualitatively, it is challenging to find the nuance to justify one ranking versus another, which makes it hard to interpret results.

## 6 Conclusion

We find that training our model using a contrastive learning task and attempting to generalize to multi-task learning is effective in improving MinBERT. However our model still does not perform as highly as others do, especially at sentiment and STS tasks.

One potential alteration to future work could be implementing a masked language modeling objective. Although this is a token-level prediction task and not a sentence-level prediction like the tasks we are trying to optimize, we think it could help our model generalize to multiple tasks by better learning language semantics and patterns.

We also could add a different normalization method to our model, suchs as Bregman proximal point approximation.

Another extension that we wish we could have implemented is using the unsupervised simCSE loss function described by Gao *et al*in their paper.[1] We are quite curious if this contrastive learning approach would have had similar results to the supervised approach that we used.

## References


[1] Tianyu Gao, Xingcheng Yao, and Danqi Chen. SimCSE: Simple contrastive learning of sentence embeddings. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2021.

[2] Emily Dinan Mohit Bansal Jason Weston Douwe Kiela Yixin Nie, Adina Williams. Adversarial nli: A new benchmark for natural language understanding. *arXiv preprint arXiv:2001.06782*, 2020.

[3] Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. *CoRR*, abs/2005.10242, 2020.

[4] Kawin Ethayarajh. How contextual are contextualized word representations? comparing the geometry of bert, elmo, and GPT-2 embeddings. *CoRR*, abs/1909.00512, 2019.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2019.

[6] Wei-Cheng Tseng. Weichengtseng/pytorch-pcgrad, 2020.

[7] Cs224n: Default final project handout.

[8] Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 632–642, 2015.

[9] Adina Williams, Nikita Nangia, and Samuel Bowman. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1804.08199*, 2018.

# A Appendix (optional)

Table 1: HyperParameter Search Results, First Round

| LR | Opt | p(drop) | Epoch | CSE-Only | CSE-Excl | Tau | Para Acc | Sent Accu | STS Corr | Total Score |
|---|---|---|---|---|---|---|---|---|---|---|
| 1e-05 | finetune | 0.3 | 1 | F | T | 0.1 | 37.77% | 14.44% | -2.11% | 16.70% |
| 1e-05 | finetune | 0.3 | 1 | F | T | 0.1 | 37.77% | 14.44% | -2.11% | 16.70% |
| 1e-05 | finetune | 0.3 | 1 | F | T | 0.1 | 37.77% | 14.44% | -2.11% | 16.70% |
| 1e-05 | finetune | 0.3 | 1 | F | T | 0.1 | 37.77% | 14.44% | -2.11% | 16.70% |
| 1e-05 | finetune | 0.3 | 1 | T | T | 0.1 | 37.80% | 14.26% | -0.93% | 17.04% |
| 3e-06 | finetune | 0.3 | 10 | F | T | 0.1 | 55.08% | 43.69% | 30.18% | 42.98% |
| 1e-05 | finetune | 0.3 | 10 | T | T | 0.1 | 42.92% | 18.53% | 6.60% | 22.69% |
| 9e-06 | finetune | 0.3 | 10 | F | T | 0.05 | 66.23% | 46.78% | 32.71% | 48.57% |
| 0.001 | pretrain | 0.3 | 1 | F | T | 0.05 | 38.63% | 34.97% | 4.42% | 26.00% |
| 0.001 | pretrain | 0.3 | 1 | F | T | 0.05 | 37.59% | 27.07% | 14.23% | 26.29% |
| 0.001 | pretrain | 0.3 | 1 | F | T | 0.05 | 46.08% | 34.15% | 10.31% | 30.18% |
| 0.0005 | pretrain | 0.3 | 3 | F | T | 0.05 | 38.99% | 25.70% | 10.73% | 25.14% |
| 0.0005 | pretrain | 0.3 | 3 | F | T | 0.05 | 38.99% | 25.70% | 10.73% | 25.14% |
| 0.0005 | pretrain | 0.3 | 3 | F | T | 0.05 | 61.59% | 30.25% | 13.17% | 35.00% |
| 1e-05 | finetune | 0.1 | 3 | F | T | 0.1 | 56.54% | 43.60% | 29.34% | 43.16% |
| 1e-05 | finetune | 0.1 | 3 | F | T | 0.5 | 55.39% | 37.60% | 30.54% | 41.18% |
| 1e-05 | finetune | 0.25 | 3 | F | T | 0.1 | 56.54% | 43.60% | 29.34% | 43.16% |
| 1e-05 | finetune | 0.25 | 3 | F | T | 0.5 | 55.39% | 37.60% | 30.54% | 41.18% |
| 1e-05 | finetune | 0.35 | 3 | F | T | 0.1 | 56.54% | 43.60% | 29.34% | 43.16% |
| 1e-05 | finetune | 0.35 | 3 | F | T | 0.5 | 55.39% | 37.60% | 30.54% | 41.18% |
| 1e-05 | finetune | 0.1 | 6 | F | T | 0.1 | 66.14% | 45.69% | 33.19% | 48.34% |
| 1e-05 | finetune | 0.1 | 6 | F | T | 0.5 | 63.85% | 46.87% | 34.35% | 48.36% |
| 1e-05 | finetune | 0.25 | 6 | F | T | 0.1 | 66.14% | 45.69% | 33.19% | 48.34% |
| 2e-05 | finetune | 0.3 | 6 | F | T | 0.1 | 71.54% | 40.42% | 30.29% | 47.42% |
| 2e-05 | finetune | 0.3 | 6 | F | T | 0.5 | 69.97% | 39.96% | 31.57% | 47.17% |
| 5e-06 | finetune | 0.3 | 10 | F | T | 0.1 | 66.35% | 46.32% | 32.21% | 48.29% |
| 5e-06 | finetune | 0.3 | 10 | F | T | 0.5 | 65.82% | 46.50% | 31.39% | 47.91% |
| 1e-05 | finetune | 0.3 | 10 | F | T | 0.1 | 70.99% | 46.59% | 31.79% | 49.79% |
| 1e-05 | finetune | 0.3 | 10 | F | T | 0.5 | 69.44% | 49.05% | 32.74% | 50.41% |
| 2e-05 | finetune | 0.3 | 10 | F | T | 0.1 | 71.96% | 40.60% | 32.42% | 48.33% |
| 2e-05 | finetune | 0.3 | 10 | F | T | 0.5 | 70.99% | 41.87% | 31.04% | 47.97% |
| 1e-05 | finetune | 0.3 | 10 | F | T | 1.0 | 69.94% | 48.32% | 32.31% | 50.19% |
| 1e-05 | finetune | 0.3 | 10 | F | T | 5.0 | 68.75% | 49.59% | 34.25% | 50.86% |
| 1e-05 | finetune | 0.3 | 20 | F | T | 0.1 | 73.99% | 50.14% | 28.46% | 50.86% |
| 1e-05 | finetune | 0.3 | 20 | F | T | 0.5 | 74.35% | 47.14% | 30.77% | 50.75% |
| 1e-05 | finetune | 0.3 | 20 | F | T | 1.0 | 72.54% | 49.59% | 28.62% | 50.25% |
| 1e-05 | finetune | 0.3 | 20 | F | T | 5.0 | 68.75% | 49.59% | 34.25% | 50.86% |
| 1e-05 | fine-tune | 0.3 | 10 | F | T | 5.0 | 70.82% | 48.86% | 34.66% | 51.45% |

Table 2: HyperParameter Search Results, Second Round

| lr | opt | p(drop) | epochs | CSE-Only | CSE-Excel | tau | opt | ParaAcc | SentAcc | STSCorr | TotalScore | PCGRAD |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1e-05 | fine-tune | 0.3 | 3 | F | T | 5.0 | ADAMW | 62.51% | 28.16 % | 7.89% | 32.85 % | T |
| 1e-05 | fine-tune | 0.3 | 0 | F | T | 5.0 | ADAMW | 66.91% | 36.33 % | 25.71 % | 42.98 % | F |
| 1e-05 | fine-tune | 0.3 | 8 | F | T | 5.0 | ADAMW | 65.43 % | 28.88 % | 12.11 % | 35.47 % | T |
| 1e-05 | fine-tune | 0.3 | 1 | F | T | 5.0 | ADAMW | 62.48 % | 25.89 % | -0.002 % | 29.40 % | T |
| 3e-05 | fine-tune | 0.2 | 8 | F | T | 5.0 | ADAMW | 64.23 % | 28.16 % | 18.70 % | 37.03 % | F |
| 3e-05 | fine-tune | 0.2 | 8 | F | T | 5.0 | ADAMW | 71.68 % | 27.16 % | 25.59 % | 41.48 % | T |
| 3e-05 | fine-tune | 0.2 | 1 | F | T | 1 | ADAMW | 62.47 % | 25.34 % | 7.31 % | 31.71 % | F |
| 3e-05 | fine-tune | 0.2 | 1 | F | T | 1 | ADAMW | 71.76 % | 39.14 % | 31.17 % | 47.54 % | T |
| 3e-05 | fine-tune | 0.2 | 8 | F | T | .1 | ADAMW | 62.47 % | 25.34 % | 18.61 % | 35.47 % | F |
| 3e-05 | fine-tune | 0.2 | 8 | F | T | .1 | ADAMW | 69.30 % | 36.34 % | 26.99 % | 44.21 % | T |
| 1e-05 | fine-tune | 0.2 | 8 | F | T | 5 | ADAMW | 68.74 % | 39.69 % | 26.96 % | 45.13 % | F |
| 1e-05 | fine-tune | 0.2 | 8 | F | T | 5 | ADAMW | 63.76 % | 28.43 % | 19.72 % | 37.30 % | T |
| 1e-05 | fine-tune | 0.2 | 8 | F | T | 1 | ADAMW | 69.36 % | 32.33 % | 28.92 % | 43.54 % | F |
| 1e-05 | fine-tune | 0.2 | 8 | F | T | 1 | ADAMW | 68.97 % | 30.43 % | 21.52 % | 40.30 % | T |
| 1e-05 | fine-tune | 0.2 | 8 | F | T | 0.1 | ADAMW | 67.49 % | 37.87 % | 29.44 % | 44.92 % | F |
| 1e-05 | fine-tune | 0.2 | 8 | F | T | 0.1 | ADAMW | 64.96 % | 27.15 % | 19.42 % | 37.18 % | T |
| 3e-06 | fine-tune | 0.2 | 8 | F | T | 5 | ADAMW | 58.71 % | 29.70 % | 08.82 % | 32.41 % | F |
| 3e-06 | fine-tune | 0.2 | 8 | F | T | 1 | ADAMW | 62.39 % | 28.52 % | 05.49 % | 32.23 % | T |
| 3e-06 | fine-tune | 0.2 | 8 | F | T | 1 | ADAMW | 62.39 % | 28.52 % | 05.49 % | 32.23 % | T |
| 3e-06 | fine-tune | 0.2 | 8 | F | T | 0.1 | ADAMW | 62.82 % | 29.88 % | 08.71 % | 33.81 % | F |
| 3e-06 | fine-tune | 0.2 | 8 | F | T | 0.1 | ADAMW | 62.57 % | 28.70 % | 06.16 % | 32.45 % | T |
| 3e-05 | fine-tune | 0.3 | 8 | F | T | 1 | ADAMW | 67.08 % | 25.98 % | 21.29 % | 38.12 % | F |
| 3e-05 | fine-tune | 0.3 | 8 | F | T | 1 | ADAMW | 71.46 % | 38.96 % | 28.30 % | 46.12 % | T |