A Comprehensive Analysis of Fine-Tuning Strategies for BERT

Stanford CS224N Default Project

Adam Chun, Emily Hsu, Emily Nguyen
Department of Computer Science
Stanford University
`{adamchun, ehsu24, eqnguyen@stanford.edu}`

Abstract

In recent years, BERT has achieved breakthrough results in the field of language modeling. Although it is pre-trained on a large corpus of data, fine-tuning BERT on a specific downstream task may be time-consuming and require significant experimentation. Our work aims to expedite this process by providing holistic insights on best practices for fine-tuning BERT. This study examines three major fine-tuning methods: (1) smoothness-inducing adversarial regularization, (2), cosine-similarity fine-tuning, and (3) multitask fine-tuning with gradient surgery. We evaluate the performance of these fine-tuning techniques on three fundamental NLP tasks - sentiment classification, paraphrase detection, and semantic text similarity (STS). We find that multitask fine-tuning with smoothness-induced adversarial regularization achieved the best overall results on the three downstream tasks and created a generalized model that does not overfit to task-specific data. Our findings contribute to a growing body of research in improving the performance of the baseline BERT model in multitask situations.

1 Key Information to include

- Mentor: David Huang
- External Collaborators: N/A
- Sharing project: N/A

2 Introduction

BERT, or Bidirectional Encoder Representations from Transformers, is a pre-trained transformer model that has enabled researchers to make groundbreaking progress on various NLP tasks such as text classification, question answering, and language translation.

Although BERT is a powerful language representation model that is pre-trained on a large corpus of data, it is not specifically designed for a particular downstream task. As a result, it may not perform optimally for a specific NLP task without further fine-tuning. With a wide toolkit for fine-tuning BERT, identifying the right approach may be time-consuming and require significant experimentation. Fine-tuning is also computationally intensive, especially for large-scale NLP tasks.

The primary motivation for this study is to examine a sample of fine-tuning techniques on three core downstream tasks in the field of language modeling: sentiment classification, paraphrase detection and semantic textual similarity (STS). In our experiment, we analyze the effectiveness of each fine-tuning technique as well as their computational tradeoffs. Our findings contribute to a growing body of research in improving the performance of the base BERT model on multiple tasks and aim to provide holistic insights on best practices for fine-tuning BERT.

3 Related Work

We first introduce the transformer model, BERT. Then, we discuss three fine-tuning methods: smoothness-inducing adversarial regularization, cosine-similarity fine-tuning, and multitask fine-tuning with gradient surgery.

The BERT model (Devlin et al., 2018) is characterized by an encoder-decoder architecture, where the encoder takes sentences as input and the decoder produces a prediction corresponding to the current task. The main mechanism that separates BERT from previous work (e.g., directional models) is its use of self-attention; this technique allows the model to learn the importance of a word by the complete context of the input sentence as opposed to a fixed window or one-way direction.

The concept of fine-tuning was introduced by Radford et al. (2018), where the authors fine-tuned GPT on tasks such as text classification and question answering. This approach demonstrated the potential of fine-tuning pre-trained models for downstream tasks. However, it has limitations in that it can be difficult to fine-tune effectively and may suffer from overfitting. To address these limitations, our work builds on three different fine-tuning strategies: smoothness-inducing adversarial regularization (Jiang et al., 2020), cosine-similarity fine-tuning (Reimers and Gurevych, 2019), and multitask fine-tuning with gradient surgery (Bi et al., 2022).

The first fine-tuning strategy is smoothness-inducing adversarial regularization (SIAR) (Jiang et al., 2020), which can improve the robustness of fine-tuning pre-trained language models by encouraging the model to produce similar outputs for inputs that are semantically similar. This is achieved by adding a smoothness regularization term to the loss function to avoid overfitting to the training data. The authors show that this regularization technique, combined with other fine-tuning methods, can significantly improve the performance and robustness of pre-trained language models on a range of natural language processing tasks.

The second approach we take is cosine-similarity fine-tuning as used by Reimers and Gurevych (2019), which uses cosine-similarity to evaluate how semantically similar two sentences are. Specifically, in sentence-BERT, cosine fine-tuning is applied to the pre-trained BERT model to generate sentence embeddings. This process involves training a linear transformation matrix on top of the BERT model so that the resulting embeddings are optimized for downstream tasks, which in our case is semantic similarity.

Finally, for multitask fine-tuning, Bi et al. (2022) contributes a novel approach to incorporating multi-field information in news recommendation models. They also apply a revised form of the gradient surgery technique to resolve gradient conflicts during multitask training. The revision aims to use the auxiliary tasks to boost the performance of the main task. We use the same gradient surgery technique in our project, but treat the three downstream tasks equally.

4 Approach

Our baseline model is the original BERT model (Devlin et al., 2018), which uses 12 Encoder Transformer layers. These layers were originally defined in the paper, Attention is All You Need (Vaswani et al., 2017). These papers describe in greater detail the architecture of the BERT model and the training/fine-tuning process, which we have diagrammed in Appendix A.1.

From a tokenized word sequence, we first used the embedding layer to convert the indices into token embeddings $\mathbf{v_1}, ..., \mathbf{v_k} \in \mathbb{R}^D$, where $D = 768$. Then, we obtained positional embeddings from learned absolute position representations (Devlin et al., 2018). Next, we implemented the Transformer layer, which is comprised of multi-head attention, an additive and a normalization layer with a residual connection, and a feed-forward layer.

For multi-head attention, we define a Q, K, and V matrix for each head, corresponding to the query, keys, and values, and compute the scaled dot-product attention (Vaswani et al., 2017), concatenating the result from each head:

$$\text{Attention}(Q, K, V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) V.$$

Then, an add and norm step is applied, which normalizes the output and relates it to the previous state, using residual connections to improve the convergence of the model. Altogether, the forward function applies: (1) the multi-head attention layer, (2) the add and norm layer, (3) a feed-forward network as defined in (Agarap, 2018), and (4) the add and norm layer again.

We also implemented the step function of the AdamW Optimizer, as described in Decoupled Weight Decay Regularization (Loshchilov and Hutter, 2017) and Adam: A Method for Stochastic Optimization (Kingma and Ba, 2014). At each timestep, the algorithm updates the first and second moment estimates and applies bias correction. The learning rate is then incorporated in the weight decay update.

Finally, we implemented the multitask classifier to train our BERT model on the following tasks: (1) sentiment classification, (2) paraphrase detection based on semantic similarity (Fernando and Stevenson, 2008), and (3) semantic textual similarity (STS) based on cosine similarity (Agirre et al., 2013).

To fine-tune the model simultaneously on all three tasks, we took two approaches. In order to apply gradient surgery, the first approach involved zipping the three datasets and taking a batch from each dataset for the corresponding task, in each iteration. We then applied gradient surgery with an out-of-the-box implementation of PCGrad, (Tseng, 2020). This approach minimizes destructive interference between conflicting gradients by projecting a task's gradient onto the normal plane of another task's gradient:

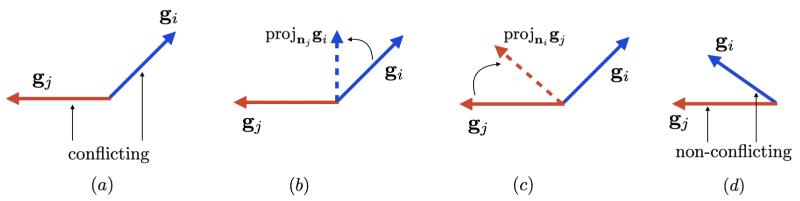


Figure 1: Visualization of PCGrad algorithm (Yu et al., 2020)

The equation for PCGrad is as follows Bi et al. (2022):

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

The second approach of multitask learning was iterating separately through the three datasets with three independent for-loops without gradient surgery. In this approach, we applied other fine-tuning methods, like smoothness-inducing adversarial regularization and cosine-similarity, as detailed below.

Smoothness-inducing adversarial regularization (SIAR) is a fine-tuning technique used to improve the generalization of models by preventing overfitting:

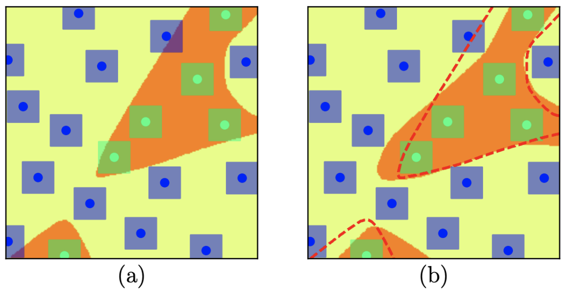


Figure 2: Decision boundaries learned without (a) and with (b) smoothness-inducing adversarial regularization, respectively (Jiang et al., 2020)

We implemented SIAR as done in the original SMART paper (Jiang et al., 2020) by introducing a regularization penalty to the loss term to result in the following optimization problem:

$$\min_\theta \mathcal{F}(\theta) = \mathcal{L}(\theta) + \lambda_s \mathcal{R}_s(\theta),$$

where $\mathcal{L}(\theta)$ is the loss function for the target task, $\lambda_s > 0$ is a tuning parameter, and $\mathcal{R}_s(\theta)$ is the smoothness-inducing adversarial regularizer, defined as

$$\mathcal{R}_s(\theta) = \frac{1}{n} \sum_{i=1}^{n} \max \ell_s(f(\tilde{x_i}; \theta), f(x_i; \theta)).$$

Here, ℓ_s is chosen as the symmetrized KL-divergence, i.e.,

$$\ell_s(P, Q) = \mathcal{D}_{KL}(P||Q) + \mathcal{D}_{KL}(Q||P).$$

To implement cosine-similarity fine-tuning, we took two approaches to obtaining embeddings for our input sentences. First, we utilized the hidden state of the [CLS] token as done in the original BERT paper (Devlin et al., 2018). Second, we took the average of the individual token embeddings to produce a fixed-length sentence embedding. With the embeddings from one of these approaches, we plugged them into the equation for the embedded sentence vectors t, e:

$$\cos(\mathbf{t}, \mathbf{e}) = \frac{\mathbf{te}}{\|\mathbf{t}\|\|\mathbf{e}\|} = \frac{\sum_{i=1}^{n} \mathbf{t}_i \mathbf{e}_i}{\sqrt{\sum_{i=1}^{n} (\mathbf{t}_i)^2} \sqrt{\sum_{i=1}^{n} (\mathbf{e}_i)^2}}$$

The output of this equation can range from -1 to 1. Therefore, since the STS dataset contains question pairs with a similarity score ranging from 0 to 5, we passed the output from cosine-similarity into a ReLU function, which clamped the score to the range [0,1]; this clamped score was scaled by 5 to match the STS scoring system.

5 Experiments

5.1 Data

For the sentiment classification task, we first used the Stanford Sentiment Treebank (SST) [1] (Socher et al., 2013), which consists of 11,855 single sentences extracted from movie reviews from which there are a total of 215,154 unique phrases mapped to one of five values: negative, somewhat negative, neutral, somewhat positive, or positive. The second dataset we used was the CFIMDB dataset, containing 2,434 highly polar movie reviews with binary positive / negative labels. For paraphrase detection, we used the Quora dataset [2], which consists of 400,000 questions pairs with labels indicating whether they are paraphrases of one another. For STS, we used the SemEval STS Benchmark Dataset (Agirre et al., 2013), which has 8,628 different sentence pairs on a scale from 0 (unrelated) to 5 (equivalent meaning).

5.2 Evaluation method

For the sentiment classification and paraphrase detection tasks, we used accuracy on the dev datasets to evaluate our model. Because these are classification tasks and/or have binary labels, accuracy is defined by the proportion of correct predictions. For the SemEval STS Benchmark Dataset, we used the Pearson correlation of the true similarity values against the predicted similarity values as our evaluation metric. Pearson correlation is calculated as follows:

$$r = \frac{\Sigma(y-m_y)(\hat{y}-m_{\hat{y}})}{\sqrt{\Sigma(y-m_y)^2 \Sigma(\hat{y}-m_{\hat{y}})^2}}$$

where y is the vector of true similarity values, $\hat{y}$ is the vector of predicted similarity values, and m_y and $m_{\hat{y}}$ are their respective means.

In addition to the task-specific accuracy, we also evaluate the fine-tuning method based on its total training time, given that some fine-tuning techniques, although effective, can have high computational trade-offs.

[1]https://nlp.stanford.edu/sentiment/treebank.html

[2]https://quoradata.quora.com/First-Quora-Dataset-Release-Question-Pairs

5.3 Experimental details

For both our baseline and multitask classifiers, we utilized the default hyperparameters for pre-training and fine-tuning. Specifically, for pre-training, we trained for 10 epochs with a hidden dropout probability of 0.3 and a learning rate of 1e-3. For fine-tuning, we also trained for 10 epochs with the same hidden dropout probability but with an adjusted learning rate of 1e-5. Due to computational costs, when utilizing the SST dataset, we made our batch size 64; for the CFIMDB dataset, we made our batch size 8. All experiments were run on AWS using the Deep Learning AMI GPU PyTorch 1.12.0 with 8 vCPUs.

We used the cosine-similarity fine-tuning method to train the model on the STS task, using the SemEval dataset. Because of the nature of this fine-tuning technique, we focused solely on the STS task for this experiment. This method is known for its efficiency and we used the default parameters, which required only 30 minutes of training time.

For smoothness-inducing adversarial regularization, we experimented with various regularization hyperparameters $\lambda_s \in \{0.1, 1, 5, 10, 20, 50\}$ across both multitask training and training just on the SST dataset with the baseline model. We used the default values for the remaining hyperparameters as specified above. Training on the baseline model took around 30 minutes, while training with multitask learning took nearly 6 hours.

For multitask fine-tuning, we experimented with and without gradient surgery, and kept the batch size (default=8) constant across the three datasets. In gradient surgery, the gradients of the three tasks are weighted equally. We also experimented with capping the training sets to the size of the shortest test set versus having no cap. The former took approximately 1 hour, while the latter took around 6 hours of training.

5.4 Results

Our results for various methods of fine-tuning are as follows:

Method	Datasets	Sentiment Acc	Para Acc	STS Corr	Training time
Baseline	SST	0.517	0.510	0.411	15 min
SMART (λ=10)	SST	0.502	0.617	0.457	30 min
Cosine-Similarity	SemEval	0.383	0.262	**0.774**	30 min
Multitask (capped)	SST, Quora, SemEval	0.500	0.691	0.751	40 min
Multitask + PCGrad	SST, Quora, SemEval	0.509	0.590	0.755	1 hr
Multitask (no cap)	SST, Quora, SemEval	0.479	0.690	0.732	6 hr
Multitask + SMART	SST, Quora, SemEval	**0.518**	**0.705**	0.726	6 hr

Table 1: Results of various fine-tuning methods

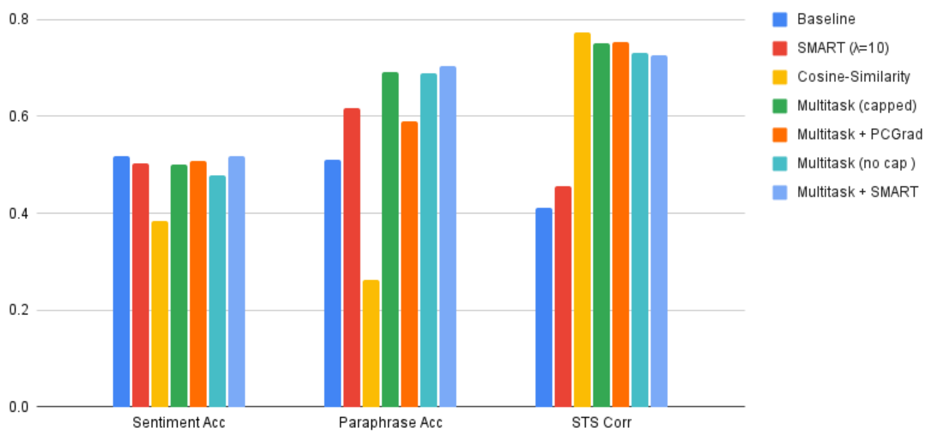


Figure 3: Visual comparison of various fine-tuning methods

Examining the results of implementing smoothness-inducing adversarial regularization on top of the baseline BERT model, we note that the model performs much stronger on the paraphrase detection task (21% increase) but does not significantly improve the other two tasks. This is consistent with the Jiang et al. (2020)'s explanation of the potential effectiveness of SIAR on various tasks, depending on the decision boundary between their outputs (to be discussed further in the Analysis section).

After experimenting with varying λ_s regularization hyperparameters, we find that the optimal λ_s for the baseline model is $\lambda_s = 10$. For small values of $\lambda_s < 5$, we find the regularization to be too weak and paraphrase detection accuracy was not maximized. For large values of $\lambda_s > 30$, regularization is too strong, resulting in decreasing sentiment accuracy and STS correlation:

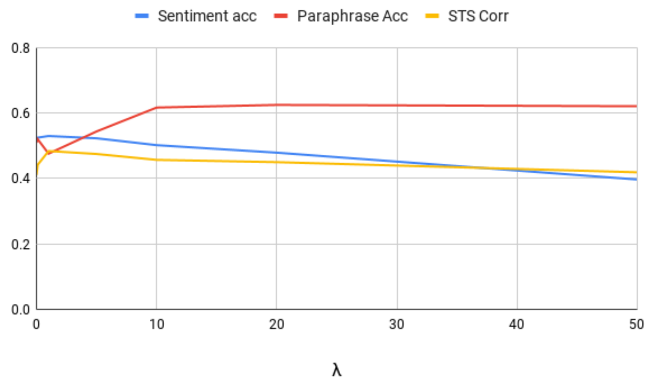


Figure 4: Visual comparison of varying regularization tuning parameters λ_s

In addition, we find that introducing cosine-similarity fine-tuning to the STS task produces the best results (STS corr=0.774) across all fine-tuning methods. This is expected because this approach has been shown to be effective in improving the performance of models like sentence-BERT Reimers and Gurevych (2019) on tasks that require measuring the similarity between two sentences. Both SIAR and cosine-similarity fine-tuning are comparable in terms of time costs, so they would be valuable additions to the fine-tuning process with low computational trade-offs.

Comparing the results between multitask with and without gradient surgery fine-tuning, we find no significant difference between the sentiment and STS tasks, and even a decrease in the paraphrase detection task performance by 15%. This may be due to the modifications of gradient surgery, which can lead to the potential loss of gradient information. Additionally, performing gradient surgery increased the training time by 50%, which is a significant trade-off with little reward.

Finally, the model produces the strongest overall results when it is trained using multitask learning and fine-tuned with SIAR. This approach created a generalized model that was not as prone to overfitting to the task-specific data. We submitted this model to the leaderboards, where it placed 88/181 on the dev set leaderboard with an overall score of 0.653, and 79/153 on the test set leaderboard with an overall score of 0.659.

6 Analysis

The two primary challenges we faced while fine-tuning BERT were (1) determining how to balance performance across three independent tasks, and (2) exchanging higher computational costs with better performance outcomes.

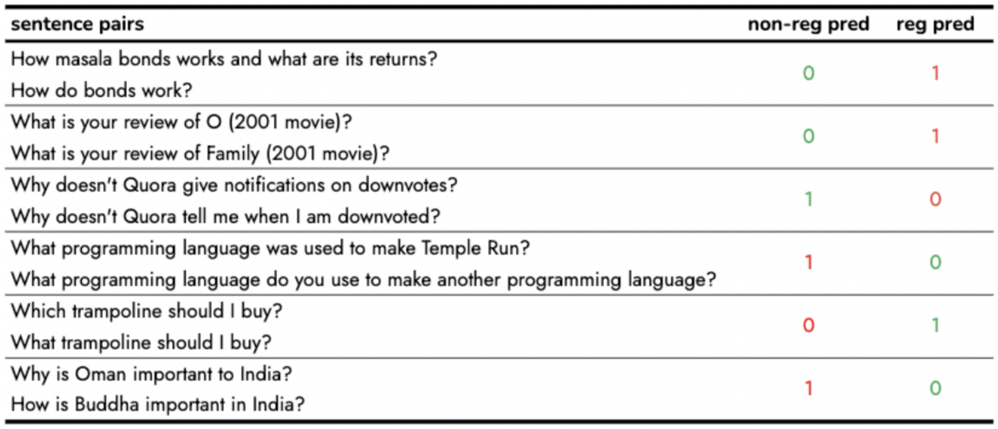
While fine-tuning for a specific task like cosine-similarity for STS is relatively straightforward and produced expected results, fine-tuning across three different tasks that may not be closely related proved to be more challenging and produced unexpected results. For example, PCGrad, which is meant to produce better results for multitask learning, proved to be ineffective in improving outcomes for our model. The shortcomings may derive from the fact that PCGrad is used to resolve conflicts between related/auxiliary tasks and boost performance for a main task (Bi et al., 2022). When PCGrad modifies the gradients to reduce destructive interference, the model may remove important gradient features related to magnitude and direction. We see this when examining the gradient conflicts of the respective tasks (Appendix A.4); resolving conflicts for sentiment classification and STS, which are more severe, may have come at the cost of stripping important information from paraphase detection and underprioritizing the task. While the model marginally improved convergence for sentiment classification and STS, the model fails to converge on the paraphrase detection task and instead, fluctuates from epoch to epoch (Appendix A.5). As a result, we find that using the out-of-the-box solution for PCGrad from Tseng (2020) is incompatible with the needs of our model, which performs three independent tasks with three different datasets.

Furthermore, our model tends to perform much better on the two tasks related to semantic similarity (paraphrase detection and STS), while performing worse for sentiment classification. This may be due to the fact that training examples for the two semantic tasks far outnumber the examples for sentiment classification. As a result, we aimed to adjust for the varying sizes of the dataset by putting a cap on all of them (see Table 1, Multitask, capped versus Multitask, no cap). This was effective in creating a more generalized model while significantly reducing training time by a factor of 6, compared to training across the full dataset which was computationally expensive.

When implementing SIAR on top of the baseline BERT model with the SST dataset, we notice that the model performs noticeably better on the paraphrase detection task (21% increase) but does not significantly improve the other two tasks. We postulate that this is caused by the nature of the classification problems and the characteristics of their datasets. Specifically, in the case of paraphrase detection, SIAR was effective because the decision boundary between paraphrases and non-paraphrases is expected to be smooth and continuous because paraphrases are generally similar in meaning and share many common features. Therefore, the model only needs to identify small differences in meaning and expression to distinguish between them. In contrast, the decision boundary for tasks such as sentiment classification and semantic similarity may be more complex and non-linear, as the features that determine the sentiment or similarity between two sentences may be more diverse and abstract.

Yet, the best results were produced by training on the full datasets with SIAR. Training on all three datasets likely created a more generalized model, while SIAR prevented it from overfitting to the task-specific data. Read together with the analysis above, multitask training with SIAR ultimately produced the best results for the paraphrase detection task and surprisingly, the sentiment classification task as well.

To see why, we perform error analysis on sentences for each of the three tasks and compare the output predictions for the multitask training model with SIAR and without SIAR. First, we begin with paraphrase detection:



sentence pairs	non-reg pred	reg pred
How masala bonds works and what are its returns? How do bonds work?	0	1
What is your review of O (2001 movie)? What is your review of Family (2001 movie)?	0	1
Why doesn't Quora give notifications on downvotes? Why doesn't Quora tell me when I am downvoted?	1	0
What programming language was used to make Temple Run? What programming language do you use to make another programming language?	1	0
Which trampoline should I buy? What trampoline should I buy?	0	1
Why is Oman important to India? How is Buddha important in India?	1	0

Table 2: Results of Non-regularized vs Regularized Predictions

Looking at the predictions by our regularized and non-regularized models in the paraphrase detection task, we see that smoothness-inducing adversarial regularization encourages the model to focus on semantic similarity due to the smoother decision boundaries. However, we see a common pattern where regularization is more likely to rely on identical words as a cue for paraphrase detection when the sentences have a low degree of semantic similarity. For example, in the second sentence pair "What is your review of O (2001 movie)?" and "What is your review of Family (2001 movie)?", the sentences are syntactically similar but semantically distinct, as they share the common words "review" and "movie." This misclassification may have occurred because SIAR placed too much emphasis on identical words. On the other hand, from the last three sentence pairs, we see that the regularized model is more effective at identifying paraphrases when the sentences have a higher degree of semantic similarity and share a larger number of identical words. Overall, SIAR performs better than the non-regularized model in paraphrase detection because the model is able to learn more robust representations of sentence meaning, leading to better generalization.

SIAR also performs slightly better than the unregularized model for sentiment classification, as shown in Table 1. Appendix A.2 details some correct classifications of the regularized model, as well as some incorrect classifications. In the first three rows, SIAR may be mistakenly picking up on more complex meanings of the language and misinterpret the movie review as sarcasm, while the unregularized model picks up the sentiments of words such as "heaviest" and "joyless." From the last three examples, we see that an unregularized model may rely too heavily on individual words or phrases, without considering the broader context. On the other hand, SIAR performs better in these cases because it encourages the model to focus more on the overall sentiment of the sentence, rather than simply matching on individual words or phrases.

For STS, the regularized and unregularized models were nearly identical in accuracy. Appendix A.3 further demonstrates correct and incorrect predictions of both models. For the first three sentence pairs, the unregularized model likely performs better because the sentences are very syntactically similar and share many identical words — for example, SIAR predicts a lower correlation for "Singapore stocks end up 0.26 percent" and "Singapore stocks end up 0.11 pct." As a result, the lexical and syntactic cues are strong indicators of similarity, so the model does not need to rely on more complex semantic representations. For the last three sentence pairs, the regularized model likely performs better because the sentences have more semantic and structural differences; SIAR predicts a lower correlation for "Yes, there is a rule against this." versus "There's no rule against it." From these examples, if the sentence pairs are very similar and share many identical words, an unregularized model may perform better because it's more prone to overfitting to these similarities and thus is able to capture more nuanced differences. However, if the sentences have more semantic differences and do not share as many identical words, SIAR may better capture the underlying relationship.

7 Conclusion

After training across three datasets with multiple fine-tuning methods, we obtained highs of 0.525 for sentiment accuracy, 0.756 for paraphrase accuracy, and 0.774 for semantic correlation. We found that using different fine-tuning approaches can lead to significant improvements in model performance. Specifically, multitask fine-tuning improved upon baseline performance for both paraphrasing and semantic similarity tasks, while smoothness-inducing adversarial regularization and cosine-similarity fine-tuning methods were effective for sentiment classification and semantic text similarity tasks, respectively.

Throughout the project, we learned about the BERT model architecture, fine-tuning techniques, and how to evaluate and compare different models. Our achievements include implementing and fine-tuning a state-of-the-art language model on multiple sentence-level tasks, and achieving competitive results on benchmark datasets. We also successfully compared and evaluated multiple fine-tuning methods and their impact on model performance.

The primary limitation of our work is the focus on only three sentence-level tasks and a limited number of datasets. Given the expensive training time of some of the fine-tuning techniques, we also weren't able to rigorously test many combinations of strategies or thoroughly experiment with hyperparameters. Thus, for future work, we hope to further investigate different combinations of techniques as well as experiment with varying hyperparameters, such as learning rate, batch size, and the tuning parameter for adversarial regularization. We would ideally run multiple trials of the same experiment to ensure consistency of results. Finally, we could also investigate how to further optimize the model and reduce training time without sacrificing performance.

References

Abien Fred Agarap. 2018. Deep learning using rectified linear units (relu). In *arXiv preprint arXiv:1803.08375*.

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

Qiwei Bi, Jian Li, Lifeng Shang, Xin Jiang, Qun Liu, and Hanfang Yang. 2022. MTRec: Multi-task learning over BERT for news recommendation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2663–2669, Dublin, Ireland. Association for Computational Linguistics.

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

Samuel Fernando and Mark Stevenson. 2008. A semantic similarity approach to paraphrase detection. In *Proceedings of the 11th annual research colloquium of the UK special interest group for computational linguistics*, page 45–52.

Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Tuo Zhao. 2020. SMART: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

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

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

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

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Ng Andrew Y Manning, Christopher D, 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*, page 1631– 1642.

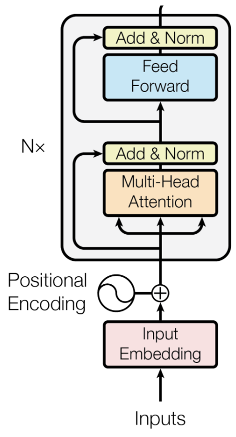
Wei-Cheng Tseng. 2020. Weichengtseng/pytorch-pcgrad.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, page 5998–6008.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. 2020. Gradient surgery for multi-task learning. *arXiv preprint arXiv:2001.06782*.

A Appendix

A.1 Encoder layer of Transformer used in BERT (Vaswani et al., 2017)



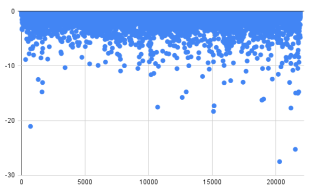
A.2 Results of Non-regularized vs Regularized Prediction on Sentiment Classification

sentences	non-reg pred	reg pred
... perhaps the heaviest , most joyless movie ever made about giant dragons taking over the world .	0	4
It would be hard to think of a recent movie that has worked this hard to achieve this little fun .	0	4
Don't waste your money.	1	3
... the last time I saw a theater full of people constantly checking their watches was during my SATs .	4	0
The best way to hope for any chance of enjoying this film is by lowering your expectations .	3	0
Whether seen on a 10-inch television screen or at your local multiplex , the edge-of-your-seat , educational antics of Steve Irwin are priceless entertainment.	1	4

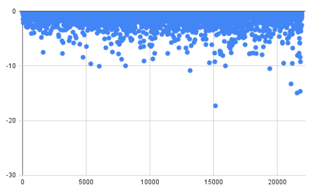
A.3 Results of non-regularized vs. regularized prediction on STS

sentence pairs	non-reg pred	reg pred
China yuan strengthens to 6.2689 against USD Chinese yuan weakens to 6.2816 against USD	0.507357657	0.6849455833
Singapore stocks end up 0.26 percent Singapore stocks end up 0.11 pct	0.6954129338	0.5281453133
China yuan weakens to 6.1559 against USD China yuan weakens to 6.1818 against USD Tuesday	0.7150636315	0.5920177698
Yes, there is a rule against this. There's no rule against it.	0.6959695816	0.5975920558
Pro-Palestinian Activists March to UN Headquarters Pro-Palestinian activists prepare flotilla to break Gaza blockade	0.6353350282	0.5397107005
Chinese shares close lower Wednesday Chinese shares close higher Friday	0.7055732012	0.6125737429

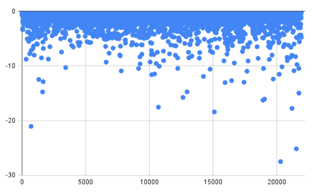
A.4 Gradient pair conflicts for (a) sentiment classification, (b) paraphrase detection, and (c) STS measured by calculating the dot product during gradient surgery.



(a)

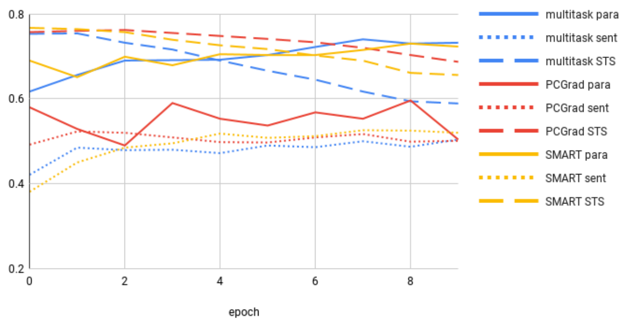


(b)

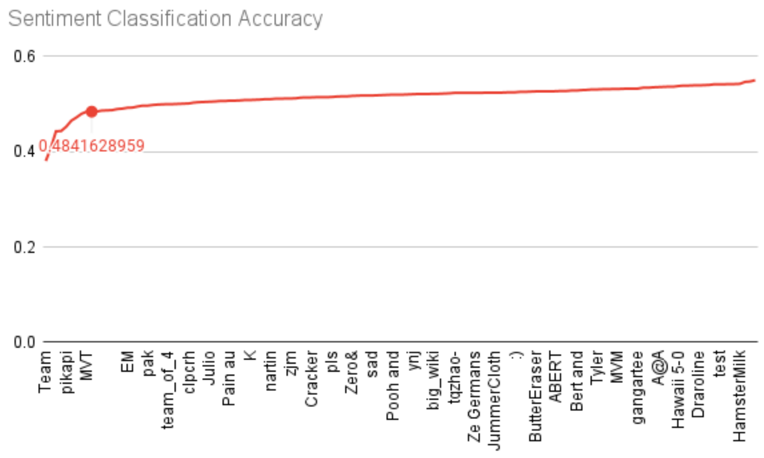


(c)

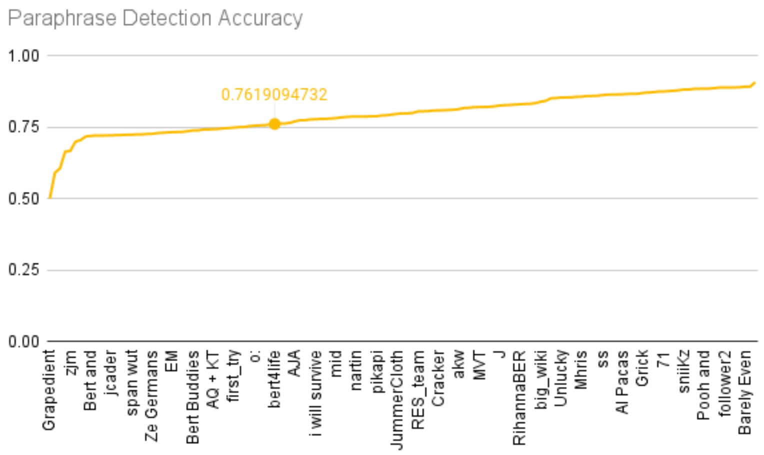
A.5 Model convergence by epoch



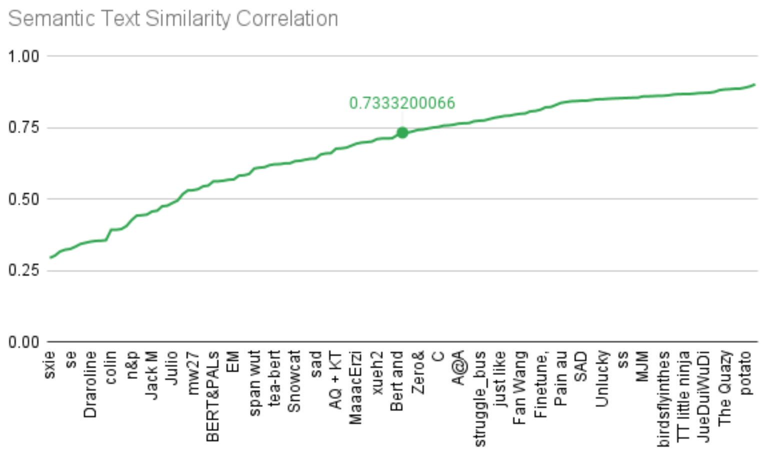
A.6 Leaderboard results for sentiment classification



A.7 Leaderboard results for paraphrase detection



A.8 Leaderboard results for STS


Semantic Text Similarity Correlation
sxie
se
Draroline
colin
n&p
Jack M
Julio
mw27
BERT&PALs
EM
span wut
tea-bert
Snowcat
sad
AQ + KT
MaaacErzi
xueh2
Bert and
Zero&
C
A@A
struggle_bus
just like
Fan Wang
Finetune,
Pain au
SAD
Unlucky
ss
MJM
birdsflyinthes
TT little ninja
JueDuiWuDi
The Quazy
potato