BERTolomeu: Exploring Methods to Improve Downstream Task Performance with BERT

Stanford CS224N Default Project

Flora Yuan
Department of Data Science
Stanford University
`floray@stanford.edu`

Jack Zhang
Department of Computer Science
Stanford University
`jyxzhang@stanford.edu`

Abstract

In the early 15th century, Bartolomeu Dias charted new waters and unveiled countless uncharted territories. Just like Bartolomeu, we navigate the vast expanse of natural language processing with BERTolomeu, seeking to optimize the multitask performance of pre-trained BERT for three downstream tasks: sentiment analysis, paraphrase detection, and semantic textual similarity. We experimented with various approaches to fine-tune the provided base BERT model on three downstream tasks simultaneously and analyzed the effectiveness of different combinations of fine-tuning techniques. We found the most success with adopting an annealed sampling approach, cross-encoding for the paraphrase detection and semantic textual similarity tasks, SMART regularization for sentiment analysis to reduce overfitting, and training on a combination of the provided and additional data. We achieve an overall score of 0.782 on the `dev` set, and 0.789 on the `test` set.

1 Key Information to include

Our mentor was Johnny Chang. We had no external collaborators, and we are not sharing this project.

Contributions: Flora owns the development of the BERT$_{\text{BASE}}$ model and additional datasets. Jack owns conducting experiments and development of sampling strategies, encoding strategies, and SMART regularization. We contributed equally to training/design strategies, SMART regularization, and writing the paper.

2 Introduction

In recent years, large pre-trained language models, such as BERT by Devlin et al. (2019) and GPT by Brown et al. (2020), have proven the effectiveness of transformer models and have driven significant advancements in the field of natural language processing (NLP). However, these pre-trained transformer models have limitations. For real-world applications, they require fine-tuning to perform specific downstream tasks and struggle with performing multiple tasks simultaneously.

We present our approach to fine-tune the pre-trained BERT model for multitask performance on three downstream tasks: fine-grained sentiment classification, paraphrase detection, and semantic textual similarity (STS). The approaches we incorporated include annealed sampling with novel tuning parameters for efficient training on datasets of varying sizes, cross-encoding for sentence-pair tasks, SMART regularization, and additional datasets to supplement tasks with smaller provided datasets. Using pretrained BERT parameters as a starting point, our aim is to determine the combination of fine-tuning approaches that yields the highest overall multitask accuracy. We present our approaches, experiments, and results to demonstrate the effectiveness and limitations of our model.

3 Related Work

Since Vaswani et al. (2023) proposed the transformer architecture in 2017, many language models proposed subsequently adopted the transformer for its superior performance. Proposed by Devlin et al. (2019), the Bi-directional Encoder Representations from Transformers (BERT) model is one such pre-trained model. Following the success of BERT, a surge of research aimed to enhance fine-tuning on downstream tasks for pre-trained models ensued.

A challenge in fine-tuning pre-trained language models for multitask performance is the imbalance in dataset sizes. When dataset sizes vary significantly, the model may overfit on smaller datasets and undertrain on larger ones. To address issue, sampling approaches, like annealed sampling by Stickland and Murray (2019), have been introduced. Annealed sampling adjusts the sampling probabilities of datasets relative to dataset size and current epoch, giving larger datasets higher sampling probabilities near the start of training and lower probabilities near the end of training.

Paraphrase detection and STS are sentence-pair tasks, meaning the model needs to learn the interdependencies between two input sentence sequences. Various encoding strategies have been proposed for sentence-pair tasks. Along with the BERT model, Devlin et al. (2019) also proposed an encoding approach for tasks involving multiple inputs called cross-encoding. In cross-encoding, input sentences are concatenated with a separator token `[SEP]` prior to passing through the model. Another encoding approach called bi-encoding have been introduced by Reimers and Gurevych (2019). Bi-encoding feeds each input sentence separately through the model, then combines their the outputs for downstream tasks. Bi-encoding has been shown to be computationally efficient and effective for similarity-based tasks.

Another prominent concern when fine-tuning large pre-trained models on relatively small datasets is overfitting. To alleviate overfitting, regularization schemes have been proposed, like SMART by Jiang et al. (2020). SMART alleviates aggressive updates by incorporating smoothness-inducing adversarial regularization and Bregman proximal point optimization, which has shown to achieve better generalization to unseen data.

4 Approach

We created our extensions to extend the pre-trained BERT model. The provided model follows a similar structure as discussed in Devlin et al. (2019).

4.1 Baseline

For our baseline, we kept the pretrained BERT parameters frozen except the last linear layer, i.e. the task specific layers. We trained the task specific layers on the full provided datasets, and evaluated the resulting performance on the `dev` set.

4.2 Sampling Strategies

The three provided datasets showed significant imbalance in size. There are 283,003 `train` examples in the Quora dataset, 8,544 in the SST-5 dataset, and 6,040 in the SemEval STS Benchmark dataset. The model can overfit on tasks with significantly larger datasets and undertrain on smaller ones. Thus, we investigated two approaches that aim to optimize the multitask learning process with varying dataset sizes.

Round-robin is a multitask sampling strategy that simply cycles through the tasks. Since there are 3 tasks, we train a batch from task 1 on training step 1, train a batch from task 2 on step 2, train a batch from task 3 on step 3, train a batch from task 1 on step 4, and so on. When a dataset is exhausted, we start over from the beginning of the dataset. The model will see smaller datasets more than once, and may not see the entirety of large datasets, alleviating the dataset size disparity.

Annealed sampling is a dynamic sampling strategy that adjusts the probability of selecting examples based on the relative size of the datasets and the current epoch. The idea is to train on tasks with abundant data more frequently in the beginning of training and gradually less frequently as training progresses, since overfitting tends to be worst near the end of training. Specifically, we select one task to sample a batch and train on at each training step. We select task i with probability p_i, which is proportional to N_i, the number of training examples for task i:

$$p_i \propto N_i^{\alpha},\ \alpha = 1 - 0.8\frac{e-1}{E-1},$$

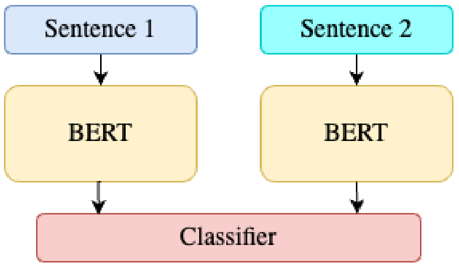
where α changes with each epoch e, E is the total number of epochs. α forces p_i for different tasks to be more equal near the end of training. We wanted to explore the effectiveness of more forceful functions for α, which is why we introduced σ as a tuning parameter:

$$\alpha = 1 - \sigma\frac{e-1}{E-1},$$

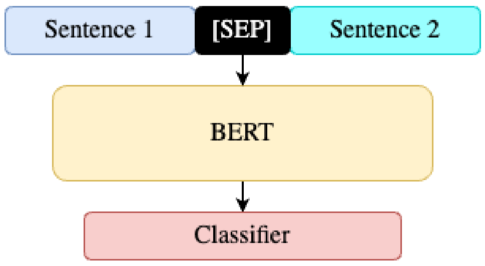
where σ defaults to 0.8. We performed a simple grid search to find the optimal value $\sigma = 0.99$ (see Appendix A).

4.3 Encoding Strategies

Paraphrase detection and and STS are sentence-pair tasks, meaning the model needs to learn the nuanced dependencies between two phrases. Yet, BERT traditionally processes single-input tasks. To address this challenge, we explored two encoding strategies: bi-encoder and cross-encoder.



(a) Bi-encoder with linear layer.



(b) Cross-encoder with linear layer.

Figure 1: Encoding strategies for sentence-pair input tasks.

4.4 SMART Regularization

The primary purpose of utilizing SMART Finetuning is to prevent overfitting and alleviate aggressive updates on multi-task data. This approach is described in Jiang et al. Jiang et al. (2020). As described in the paper, this technique incorporates Smoothness-Inducing Adversarial Regularization and Bregman Proximal Point Optimization.

Smoothness-Inducing Adversarial Regularization aims to enforce smoothness by minimizing the local Lipschitz continuity of the model, which is achieved by introducing small perturbations (ϵ) to the input data and optimizing Equation 1 in the paper

$$min_{\theta}\mathcal{F}(\theta) = \mathcal{L}(\theta) + \lambda_s\mathcal{R}_S(\theta) \tag{1}$$

where $\mathcal{L}$ represents the loss function and $\mathcal{R}_S(\theta)$ represents the smoothness-inducing adversarial regurlarizer (full equations definded in paper).

Bregman Proximal Point Optimization, used in conjunction with the adversarial regularizer, serves as a strong regularizer by preventing the model parameters from deviating significantly between iterations. This method is implemented through momentum Bregman Proximal Point method (MBPP), which is shown in Equation 3:

$$\theta_{t+1} = \arg\min_{\theta} \mathcal{F}(\theta) + \mu\mathcal{D}_{\text{Breg}}(\theta, \tilde{\theta}_t) \tag{2}$$

However, in our implementation, we used the ADAMW optimizer to solve the minimization, instead of their proposed Bregman Proximal Point method. This effectively retains the knowledge of the pre-trained model while adapting to new tasks. The combined approach of SMART fine-tuning not only enhances the robustness of the model but also optimizes its performance across various NLP tasks.

4.5 Additional Data

Another approach we attempt to alleviate the disparity in dataset sizes is to seek additional datasets for training.

For the sentiment analysis task, we are evaluated on the Stanford Sentiment Treebank (SST-5), which is a dataset of movie reviews. Due to the unique nature of this genre of writing that feature highly polarized opinions and nuanced expression of these opinions, we searched for similar datasets from movie review websites like RottenTomatoes and IMDB. We decided to complement this task with the Large Movie Review Dataset (LMRD) from Maas et al. (2011), a collection of 50,000 highly polarized reviews from IMDB since it is most similar to SST-5.

We decided to not look for any additional data for the paraphrase detection task, as the provided QQP dataset was already significantly larger than the provided SST and SemEval datasets.

For STS, we decided to complement this task with an additional dataset called Sick2014 Marelli et al. (2014). This dataset is similar to SemEval STS Benchmark, but aims instead to capture only similarities on purely language and common knowledge level, without relying on domain knowledge like history. Sick2014 contains 10,000 English sentence pairs, each annotated for relatedness in meaning.

To couple the increased data, we experimented with increasing the number of epochs. By extending the training duration, we aimed to give our model more opportunities to learn and generalize from the available data. This allows the model to make multiple passes over the data, refining its understanding and improving its ability to capture complex patterns.

5 Experiments

5.1 Data

We have three datasets provided by the CS224N staff, and two additional datasets we use to alleviate size disparity among the provided datasets. All provided datasets come with respective `train`, `dev`, and `test` splits, and we create two new `train` splits for datasets that mix provided and additional data.

For **sentiment analysis**, the SST-5 dataset is provided, and the LMRD is additional Maas et al. (2011). SST-5 have 8,544 `train` examples, 1,001 `dev` examples, and 2,210 `test` examples. LMRD has 50,000 examples. The format of both datasets is the same, both consisting of phrases extracted from movie reviews. Each phrase is either labeled as negative, somewhat negative, neutral, somewhat positive, or positive (on a respective scale from 0 to 4). When we train on additional data, we use a combination of the two datasets. To create a balanced combined dataset, we concatenate randomly sampled 60% of the data from the SST-5 dataset and 40% from the LMRD dataset, then shuffled the entries to eliminate order bias. The result is a combined dataset with 21,126 `train` examples. We keep the provided 1,101 `dev` examples and `test` examples.

For **paraphrase detection**, we use the Quora Question Pairs (QQP) dataset, which is made up of question pairs and indicates whether the sentences are paraphrases of each other. We have 283,010 `train` examples, 40,429 `dev` examples, and 80,859 `test` examples.

For **semantic textual similarity**, the SemEval STS Benchmark dataset is provided and the Sick2014 dataset is additional. SemEval STS Benchmark has 6,040 `train` examples, 863 `dev` examples, and 1725 `test` examples. Sick2014 has 10,000 examples. The format of both datasets is the same, consisting of sentence pairs of varying similarity from 0 (not related at all) to 5 (equivalent). Similar to sentiment analysis, we create a combined dataset with 5,424 `train` examples, containing 60% of the provided data and 40% of the additional data randomly sampled and shuffled.

5.2 Evaluation method

To evaluate our model, we are using the task-specific metrics provided in the default final project handout. For the combined sentiment (SST-5 and IMDB) and paraphrase detection (QQP), we are evaluating performance based on the accuracy between the true and predicted labels. For the combined semantic textual similarity (SemEval STS and Sick2014), we are evaluating performance based on the Pearson correlation of the true similarity values against the predicted similarity values.

5.3 Experimental details

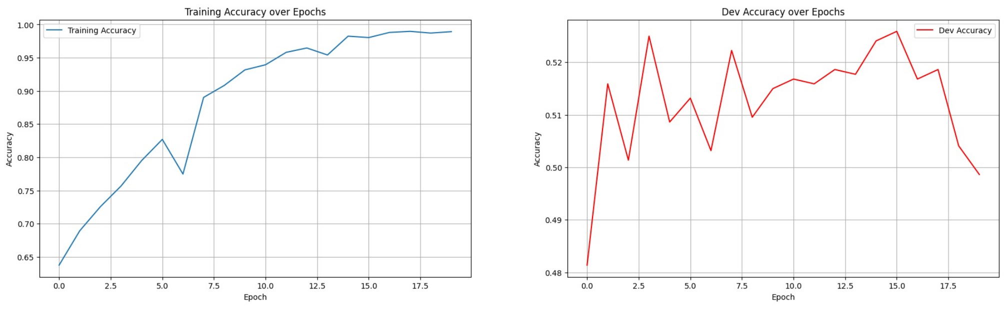
In this section, we detail the explorations we made to find the best performing combination of extensions. All experiments used the $\text{BERT}_{\text{BASE}}$ model with provided pre-trained parameters. Whenever we finetune with all BERT layers frozen except the last task-specific layer, we use a learning rate of 1e-3, which we call the "last linear layer" mode. When we train to update all model parameters, we use a learning rate of 1e-5, which we call the "full model" mode. All experiments used a hidden layer dropout probability of 0.3, a batch size of 8, and 4096 training steps per epoch. We also used the default AdamW optimizer hyperparameters. For the loss function, we use cross entropy for sentiment classification, binary cross entropy for paraphrase detection, and mean squared error (MSE) loss for STS. We will describe how each of the following experiments with our extensions builds on top of each other.

Sampling Strategies: Our first experiment compared the two sampling strategies. We finetuned the full model first with round-robin sampling. After that, we finetuned the full model with annealed sampling. As shown in Table 1, annealed sampling outperformed round-robin in all three tasks. Furthermore, as shown in Table 3, annealed sampling with $\sigma = 0.99$ yielded the highest overall `dev` accuracy.

Cross-Encoding: To address the outstandingly low STS correlation after our first experiment, we explored the performance of another encoding strategy. We compared two encoding approaches: bi-encoding and cross-encoding. In bi-encoding, we pass sentence pairs individually through BERT, concatenate the outputs, and then apply a linear classification layer. In cross-encoding, we concatenate the sentence pair with a [SEP] token before passing through BERT and linear classification layer.

As shown in Table 1, cross-encoding significantly improved performance for both STS and paraphrase detection, since the model can better learn the relationship between two input sentences. Hence, we decided to use cross-encoding in our final model for tasks with sentence-pair inputs.

SMART: We observed signs of overfitting for sentiment analysis, where `train` accuracy was increasing while `dev` accuracy was decreasing. Thus, for our third experiment, we implemented SMART regularization specifically for sentiment analysis to reduce overfitting. As shown in Table 1, accuracy improved for the sentiment task without compromising performance on other tasks. A drawback is SMART significantly increased the training time, since we double the number of passes through BERT (once unperturbed, and once with noise) for the task.



(a) Sentiment `train` accuracy over training epochs. (b) Sentiment `dev` accuracy over training epochs.

Figure 2: The sentiment classification task shows signs of overfitting between epochs 15 and 20, as the `train` accuracy is increasing while the `dev` accuracy is decreasing. This implies the model is getting better at predicting the `train` data, but getting worse at predicting `dev` data, a sign of losing generality. The model was trained on full model for 20 epochs, with round robin sampling, cross encoding, and SMART.

Additional Data: To complement dataset size for sentiment analysis and STS and support model generalization, we constructed mixed datasets that randomly samples provided and additional data at a 60-40 ratio, i.e. 60% provided data and 40% additional data. We finetuned full model on the mixed datasets (leaving QQP unchanged) for 20 epochs, then finetuned last linear layer on the provided datasets for 3 epochs, since we want the model to see just the provided data before evaluation. As per Table 1, training on the mixed datasets improved accuracy for the two targeted tasks. The training time increased notably because the IMBD dataset contains movie reviews that are on average longer in length compared to its provided dataset counterpart: the average number of characters per movie review is 1285.49 for the IMBD dataset, and 103.74 for the SST-5 dataset. We saw further improvement after finetuning on last linear layer with just the provided data.

Number of Epochs: We trained for 10 epochs for most of our early experiments. We later discovered signs of underfitting and increased the total epoch count to 20. Upon this switch, we saw clear signs of overfitting as in Figure 2, which shows that the optimal total epoch for the sentiment analysis task is 15 epochs. Other tasks did not show any sign of overfitting, so we continued with 15 epochs for training.

5.4 Results

After all our experiments, we found the best combination of extensions is as follows: annealed sampling with $\sigma = 0.99$, cross encoding for paraphrase detection and STS, SMART regularization in sentiment analysis to mitigate overfitting, and additional data to improve model generalization. This strategy allowed us to surpass our baseline by a significant amount.

For convenience and spacing sake, we will label many epochs we ran with for each of the approaches. Unless otherwise specified, we train for 10 epochs. We will mark changes in epochs with symbols. "#" represents training on full model for 15 epochs, and "$*$" represents 20 epochs. "$" represents training on the last linear layer for 3 epochs on top of results from the full-model approach in the cell directly above.

Table 1: `Dev` performance of multitask models on downstream tasks. All models are trained on a server with Intel Skylake 2 vCPU 7.5 GB memory, 1 NVIDIA T4 GPU.

Model	Overall Score	Sentiment Accuracy	Paraphrase Accurary	STS Correlation	Training Time
Baseline	0.563	0.401	0.702	0.172	02:30
RR	0.642	0.500	0.755	0.339	**01:50**
AS	0.662	0.506	0.780	0.400	02:05
RR + CE	0.764	0.499	0.855	0.874	02:00
AS + CE	0.774	0.500	0.885	0.872	02:00
RR + CE + SMART	0.737	0.513	0.764	0.870	05:00
RR + CE + SMART (*)	0.768	0.489	0.876	0.878	03:30
RR + CE + SMART + AD (*)	0.773	0.504	0.881	0.866	16:40
AS + CE + SMART (*)	0.780	0.502	0.898	**0.879**	03:30
AS + CE + SMART + AD (#)	0.780	0.516	**0.892**	0.866	10:00
AS + CE + SMART ($)	**0.782**	**0.523**	0.891	0.867	00:30

As seen in Table 1, our model significantly surpassed the baseline in all three tasks. These improvements demonstrate the model's enhanced ability to understand sentiments within text, similarity in text, and nuanced semantic relationships.

Table 2: `Test` performance of single and multitask models on downstream tasks

Model	Overall Score	Sentiment Accuracy	Paraphrase Accurary	STS Correlation
Baseline	0.563	0.401	0.702	0.172
AS + CE + SMART + AD (#$)	0.789 (+0.226)	0.541 (+0.140)	0.891 (+0.189)	0.869 (+0.697)

As shown in Table 2, our performance on the `test` set further confirms the effectiveness of our approach. We can see that the model saw a large increase in the accuracies across the board, with an especially notable increase in STS correlation of 0.697. We attribute this increase in STS to annealed sampling and cross-encoding. Our initial tests had far lower STS correlation compared to sentiment and paraphrase accuracy indicated to us that the model is significantly overfitting to the paraphrase task, due to the overwhelming difference between the sizes of the paraphrase and STS datasets. By implementing annealed sampling, we were able to ensure that the model is able to train on STS. Cross-encoding further allowed the model to learn nuanced similarities between the input sentence pairs.

6 Analysis

To better understand our best model's performance, we conducted a qualitative evaluation by looking closer at the predicted and true results. In this section, we explore our findings, focusing on where the model performs well, where the model performs poorly, and key characteristics.

6.1 Sentiment Classification

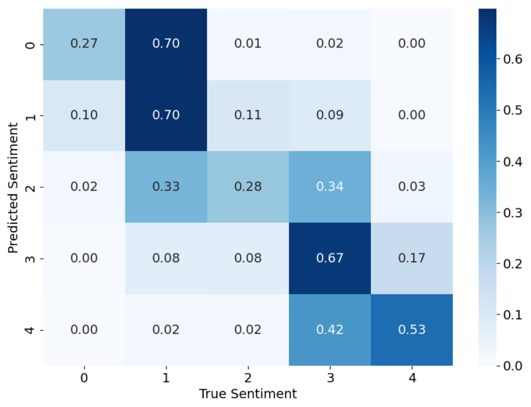


Figure 3: Normalized confusion matrix for sentiment classification

We decided to visualize our model's performance in sentiment classification using a confusion matrix (Figure 3). The diagonal values represent the proportion of correctly predicted instances for each class, while the off-diagonal values represent misclassifications. We can see that 3 out of the 4 highest accuracy blocks are on the diagonal: 1, 3, and 4. This indicates that our model is better at predicting extreme sentiments and worse at discerning neutral and weak sentiments. We believe that this is due to the composition of our additional data, namely containing only extreme sentiments (exclusion of neutrals).

Additionally, we see there is a high level of misclassification, particularly between classes 0 and 1. Specifically, 70% true class 1 are misclassified as class 0. Similarly, 42% of true class 3 instances were misclassified as class 4. These two notable misclassification examples show that there is substantial confusion between adjacent classes, suggesting that the model struggles with fine-grained sentiment distinction. We believe this is due to variations in human judgment during the labeling process, which may have impacted our model's accuracy.

6.2 Paraphrase Detection

Though examining our model's paraphrase detection performance, we discovered that our model has difficulty distinguishing sentence pairs that are highly similar in general context but have subtle differences that make them not paraphrases of each other. For example, consider the sentence pair: "How can I overcome this fear?" and "How does one overcome a fear of power?", which our model incorrectly misclassifies as paraphrases of each other. Notice that both sentences are generally talking about overcoming fear; however, they differ in details, such as the specific focus on the fear of power in the second sentence. This illustrates our model's limitations with identifying nuanced distinctions in sentence pairs that share similar structures and vocabulary but convey different specific actions or contexts.

6.3 Semantic Textual Similarity

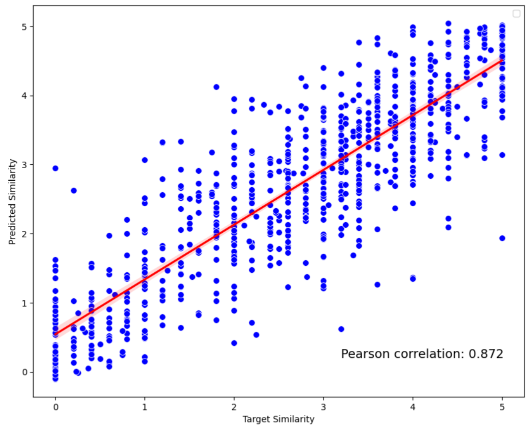


Figure 4: Scatter plot of predicted and target STS similarities

To analyze our model's performance in semantic textual similarity through a scatter plot of predicted vs target similarities (Figure 4). Our analysis revealed there is a fairly strong correlation between the predicted and target STS similarities. From the graph, we also notice outliers, representing when our model inaccurately determined the similarity of sentences with different meanings.

For example, the sentences "A woman opens a window." and "A woman is looking out a window." share many of the same key words. However, we can see that they describe two completely different actions. Despite ths, our model predicted a similarity score of 3.95, compared to the true similarity score of 2.0. This misclassification example suggests that the model places too much emphasis on surface-level lexical similarities rather than the underlying semantic differences, leading to an inflated similarity score.

7 Conclusion

In this paper, we presented a model using multitask learning strategies to enhance the $\text{BERT}_{\text{BASE}}$ model's ability to assess downstream tasks. Throughout our process, we experimented with a plethora of techniques, including round-robin, annealed sampling, bi-encoding, cross-encoding, SMART regularization, and additional data. Through our experiments, we demonstrated the effectiveness of our final model, which greatly exceeded our initial baseline model on all three tasks. Our final model incorporated annealed sampling with $\sigma = 0.99$, cross-encoding for paraphrase detection and STS, SMART regularization for sentiment classification, and additional data. We showed the strengths of our model in our analysis, highlighting its ability to classify sentiment, detect paraphrasing, and recognize semantic similarity. Additionally, we showed our model's limitations in truly understanding specific distinctions on a more nuanced level. For further research, we would look into exploring variations of the methods we have proposed, including experimenting with different splits between provided and additional data, the number of training steps, SMART regularization weight selection, and more.

8 Ethics Statement

One concern is the potential for bias in the training data, which leads to biased predictions. If the training data contains biased language or reflects societal prejudices, the model may inadvertently perpetuate or even exacerbate these biases, resulting in unfair treatment of certain groups, misrepresentation of sentiments, and incorrect paraphrase or similarity judgments that align with biased perspectives. To mitigate this, it is crucial to employ diverse datasets and monitor the model's outputs for bias. Additionally, there is a potential risk of misuse of these models. For instance, these models could be exploited to create sophisticated fake news or misinformation by slightly altering true statements. To mitigate these risks, it is important to establish clear ethical guidelines for a model's use and promote accountability by monitoring the applications and enforcing appropriate regulations.

References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

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*, pages 2177–2190, Online. Association for Computational Linguistics.

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.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014. A SICK cure for the evaluation of compositional distributional semantic models. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 216–223, Reykjavik, Iceland. European Language Resources Association (ELRA).

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Asa Cooper Stickland and Iain Murray. 2019. Bert and pals: Projected attention layers for efficient adaptation in multi-task learning.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention is all you need.

A Appendix: Optimal value for σ in Customized Annealed Sampling

Table 3: `Dev` performance of various σ values for annealed sampling in multitask models on downstream tasks. All models are trained on a server with Intel Skylake 2 vCPU 7.5 GB memory, 1 NVIDIA T4 GPU.

σ	Overall Score	Sentiment Accuracy	Paraphrase Accurary	STS Correlation
0.7	0.643	0.479	**0.785**	0.328
0.8 (default)	0.662	0.506	0.780	**0.400**
0.9	0.647	0.499	0.761	0.359
0.95	0.657	0.515	0.782	0.347
0.99	**0.662**	**0.529**	0.777	0.362
0.995	0.651	0.485	0.781	0.376

To determine the optimal value for σ in annealed sampling, which, for context, is defined as follows:

$$p_i \propto N_i^{\alpha},\ \alpha = 1 - \sigma\frac{e-1}{E-1},$$

where p_i is the probability with which we select task i, N_i is the number of training examples for task i, and α changes with each epoch e, E is the total number of epochs.

We used annealed sampling as the only extension on top of the BERT$_{\text{BASE}}$ model, and trained with various values for σ. We show the results in Table 3, $\sigma = 0.99$ yielded the highest overall score. While other σ values had higher accuracies on paraphrase detection and STS, $\sigma = 0.99$ won out overall due to significantly higher sentiment classification accuracy.