Semantic Symphonies: BERTrilogy and BERTriad Ensembles

Stanford CS224N Default Project

Yaohui Zhang
Department of Electrical Engineering
Stanford University
yhzhang3@stanford.edu

Haoyi Duan
Department of Electrical Engineering
Stanford University
haoyid@stanford.edu

Abstract

In this paper, we explore the design space of BERT model by studying how different design choices will affect the performance on three downstream tasks. Specifically, we enhance our model's representational capabilities and performance on specialized tasks through SimCSE or further pre-training before fine-tuning. We emplement single-task finetuning – **BERTrilogy**, and multi-task finetuning – **BERTriad**. In the BERTrilogy setting, we add SMART loss to mitigate the overfitting problem. In the BERTriad setting, we use the specially designed "smart-round-robin" routine to facilitate gradient surgery, which addresses the issue of conflicting gradients in multitask learning. Finally, we implement ensembling the predictions of our best models. Our best results achieve mean performance of **80.2%** on the dev set and **79.6%** on the test set, respectively.

1 Key Information to include

- Mentor: Cheng Chang
- External Collaborators (if you have any): None
- Sharing project: None

2 Introduction

The success of large pre-trained models such as BERT Devlin et al. (2018), GPT Brown et al. (2020) has proven the effectiveness of transformer architectures in various natural language processing(NLP) tasks, including text classification, question answering, machine translation, etc. However, these models have limitations.They need task-specific finetuning and is not capable of performing multi-tasks simultaneously. Multitasks learning has emerged as a promising approach to overcome these limitations by enabling models to learn from multiple tasks at once, potentially improving their generalization capabilities across a broader range of tasks.

In this project, we use pretrained embeddings from the BERT model as a starting point and try to adapt various strategies to gain better performance on downstrean tasks including **further pretraining**, **SimCSE** and **multitask learning**. Since many combinations of these approaches yields impressive results, we decide to **ensemble** the predictions of our best models in different settings to advance further, which has been proven empirically effective by Liu et al. (2019). See Figure 1 for the whole framework of our method.

This paper continues with a quick review of related work in Section § 3. Then we discuss our model approaches in Section § 4. We present our main experiments and corresponding results in Section § 5. A comprehensive analysis for experimental results is given in Section § 6. Section § 7 constitutes of our conclusion and ideas for future work.

3 Related Work

The transformer architecture was originally proposed in Vaswani et al. (2017) for neural machine translation. Its superior performance motivated Devlin et al. (2018) to propose a bidirectional transformer-based language model named BERT. Specifically, Devlin et al. (2018) pre-trained the BERT model using a large corpus without any human annotation through unsupervised learning tasks. The pre-trained language model is then adapted to downstream tasks and further finetuned. The top layer of the language model can be replaced by a task-specific layer and then continue to train on downstream tasks.

While pre-trained models demonstrate promising capability and potentials in various downstream tasks, they are tortured by the *anisotropy* problem Ethayarajh (2019), i.e., the learned embeddings are closely clustered in the vector space, which severely limits their expressiveness. Gao et al. (2019) shows that language models trained with tied input/output embeddings lead to anisotropy word embeddings, and this is further explored by Ethayarajh (2019) in pre-trained contextual representations. To mitigate this problem, post-processing techniques such as eliminating the dominant principal components Arora et al. (2017) and mapping embeddings to an isotropic distribution Li et al. (2020) have been proposed. SimCSE Gao et al. (2021) adopts a simple contrastive learning framework to obtain more aligned and uniform embeddings, which has greatly pushed the stat-of-the-art in many downstream tasks. The unsupervised approach takes an input sentence and predicts *itself* in a contrastive objective, with only standard dropout used as noise, while the supervised approach incorporates annotated pairs from natural language inference datasets by using "entailment" pairs as positives and "contradiction" pairs as hard negatives.

In the pursuit of versatile models, multitask learning Caruana (1997) serves as a powerful technique aiming to enhance the performance and generalization ability of models across multiple tasks simultaneously. Unlike traditional finetuning, which optimizes a pre-trained model on a single specific task, multitask finetuning involves adjusting the model on several related tasks at the same time, which can greatly improve model robustness and transferability. One potential risk for multitask finetuning is that it is not always benificial depending on how the model is finetuned. Gradients directions of different tasks may conflict with one another. Yu et al. (2020) recommend a technique called Gradient Surgery. If two gradients are conflicting, the gradients are altered by projecting each onto the normal plane of the other to prevent the interfering components of the gradient from being applied to the network. In addition, due to the limited data from the target task and the extremely high complexity of the pre-trained model, aggressive finetuning often makes the adapted model overfit the training data of the target task and therefore does not generalize well to unseen data. Jiang et al. (2020) introduced *Smoothness-inducing Regularization* to effectively manages the complexity of the model and *Bregman Proximal Point Optimization* to prevent aggresive updating.

4 Approach

We first complete minimal BERT implementation, adding multihead self attention module, skip connections, layer normalizations, position wise feed-forward layer and 10% dropout, based on the provided skeleton code.

To find the optimal head architecture for each downstream task, we maintained consistent hyperparameters and conducted extensive experiments with a variety of potential head architectures. This included linear layers or multi-layer perceptrons (MLPs) paired with various activation functions. For the SST task, we observed that employing a Multi-Layer Perceptron as classifier yields the most favorable results, reaching an accuracy of 53.3%. For QQP and STS, our initial intuitive implementation is calculating the cosine similarity between the two [CLS] embeddings. This method achieved an accuracy of about 80% for QQP and Pearson correlation of approximately 60% for STS, respectively. Further, instead of treating two paired sentences separately, we feed BERT a long sentence concatenated by each pair of sentences as our embedding. For QQP task, using a linear layer with ReLU as activation achieves 89.3% accuracy, while for STS task, using a linear layer with Sigmoid as its activation reaches Pearson correlation of 86.5%. Thus we established our multitask baselines, **BERTrilogy**, for Sentiment Analysis Socher et al. (2013), Paraphrase detection Fernando and Stevenson (2008) and Semantic Textual Similarity Agirre et al. (2013) by finetuning single-logit BERT instances with cross entropy, binary cross entropy and squared error losses respectively. This

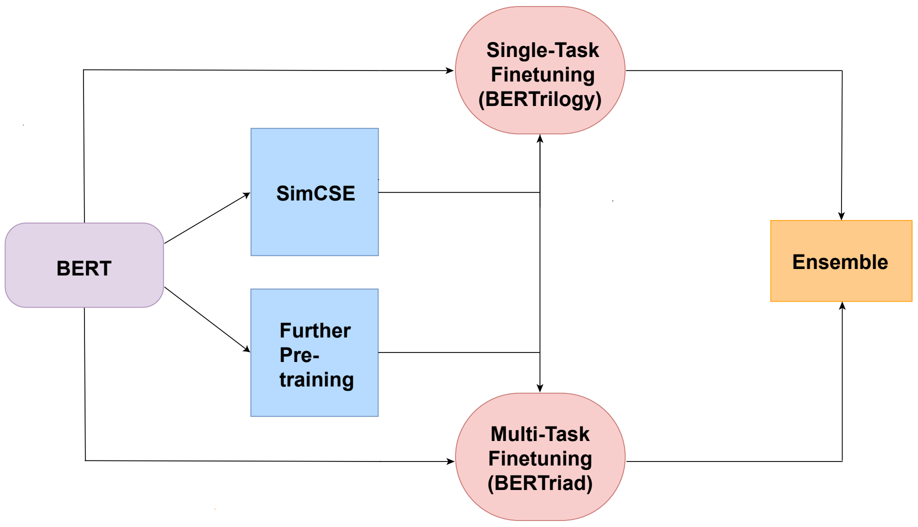
Single-Task Finetuning (BERTrilogy)
SimCSE
BERT
Ensemble
Further Pre-training
Multi-Task Finetuning (BERTriad)

Figure 1: **Framework of our method.** Before fine-tuning, we enhance the representational capabilities through SimCSE or further pre-training. After finetuning, we implement ensembling the predictions of our best models.

independent finetuning strategy has an advantage of allowing independent extension designs specific to each dataset.

However, BERTrilogy has a disadavantage of more storage and less generalization, we further design **BERTriad** – using round-robin and gradient surgery to train three tasks simultaneously.

Here are our extensions for BERTrilogy and BERTriad:

SimCSE. Learning universal sentence embeddings is a fundamental problem in natural language processing and has been studied extensively in the research (Kiros et al. (2015); Hill et al. (2016)). It is widely known that though pre-trained embeddings have good alignment, their uniformity is poor (i.e., the embeddings are highly anisotropic) Wang and Isola (2020).

Gao et al. (2021) introduces a simple contrasive learning framework that greatly advances state-of-the-art sentence embeddings, especially in Semantic Textual Similarity (STS) tasks. Specifically, the contrastive loss function here encourages embeddings of similar samples to be close together, while pushing embeddings of dissimilar samples further apart in the embedding space. It assumes a set of paired examples $\mathcal{D} = \{(x_i, x_i^+)\}_{i=1}^m$, where x_i and x_i^+ are semantically related. The contrastive framework in Chen et al. (2020) takes a cross-entropy objective with in-batch negatives Henderson and Liu (2017): let h_i and h_i^+ denote the representations of x_i and x_i^+, the training objective for (x_i, x_i^+) with a mini-batch of N pairs is:

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

where τ is a temperature hyperparameter and $sim(h_1, h_2)$ is the cosine similarity $\frac{h_1^\top h_2}{\|h_1\| \cdot \|h_2\|}$. We encode input sentences using BERT Devlin et al. (2018): $h = f_\theta(x)$, and then fine-tune all the parameters using the contrastive learning objective (Eq. 1). The *unsupervised* version of SimCSE tries to predict the input sentence itself with only dropout Srivastava et al. (2014) used as noise, while the *supervised* version builds upon natural language inference (NLI) datasets Conneau et al. (2018) and incorporates annotated sentence pairs in contrastive learning. As illustrated in Figure 1, We employ SimCSE method before finetuning.

Further Pre-training. To customize and enhance our model's performance on specialized tasks, we introduce separate further pre-training for both Sentiment Analysis Task and Paraphrase Detection/Semantic Textual Similarity Task (Figure 1). Given that the training corpora are inevitably imbalanced, the BERT model's capability to understand text from different fields can vary significantly. Our intuition is that by further pre-training our model in task-specific corpora, it would enable our model comprehend input sentences better, such as movie reviews in Sentiment Analysis Task, and results in Figure 3 validates our assumption. For Sentiment Analysis Task, we use IMDB 50K dataset[1], which is a comprehensive collection of movie reviews designed to facilitate sentiment analysis research. We simply extract the input sentences for pre-training purpose (MLM task introduced by Devlin et al. (2018)) without finetuning on it. For Paraphrase Detection/Semantic Textual Similarity Task, we treat both of them as sentence-pair related tasks and do further pre-training on SNLI[2] Corpus follow the same step as previous task.

SMART Loss. While the pretraining - finetuning paradiam has fundamentally changed the landscape of natural language processing (NLP), aggressive finetuing often causes the finetuned model to overfit the training data of downstream tasks and fail to generalize to unseen data due to limited data resources from downstream tasks and the extremely high complexity of pretrained models. To address such an issue in a principled manner, Jiang et al. (2020) propose a new learning framework for robust and efficient fine-tuning for pre-trained models to attain better generalization performance. The essence of SMART Loss includes regularization in the finetuning loss function and parameter gradient step. By introducing our "adversaries" as our input (in this case, embeddings) with small perturbations (randomly generated noise) added to it, it helps the model to be smoother with respect to the input data. We penalize the model when its output changes significantly with these added perturbations.

Instead of minimizing the typical loss $\mathcal{L}(\theta) = \frac{1}{n}\sum_{i=1}^{n} \ell(f(x_i;\theta), y_i)$ for some task-specific ℓ (in our case cross entropy for SST and QQP, squared error for STS), they minimize $\mathcal{L}(\theta) + \lambda R_s(\theta)$, where $R_s(\theta) = \frac{1}{n}\sum_{i=1}^{n} \max_{\|\tilde{x}_i - x_i\|_p \leq \epsilon} \ell(f(x_i;\theta), f(\tilde{x}_i;\theta))$ for some sampled embedded input perturbation $\tilde{x}$. We use the ADAMW optimizer to solve this minimization instead of adopting their Bregman Proximal point method.

Gradient Surgery. We addressed the issue of conflicting gradients in multitask learning by implementing gradient surgery using PCGrad, a technique to mitigate negative transfer between tasks Yu et al. (2020). This method projects the gradient of the i-th task g_i onto the normal plane of another conflicting task's gradient g_j, depicted mathematically below.

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

We used the PyTorch implementation provided by Tseng (2020).

Round-Robin. We implement three kinds of round-robin routines to make BERTriad generalizable across different tasks. The first one is "cycle-round-robin", which iteratively cycles through smaller datasets, enabling each task to generate logits within every batch iteration. This method supports gradient surgery becuase it can manipulate the gradients of all three tasks at the end of each batch. However, "cycle-round-robin" encounters overfitting problems. The second routine, "shuffle-round-robin", processes a random batch from one of the three dataloaders in each batch iteration. This technique can not support gradient surgery. We further introduce the third approach: "smart-round-robin." This method involves randomizing the process of smaller datasets in proportion to the largest dataset during each batch iteration, thereby accommodating gradient surgery and avoiding overfitting. The "smart-round-robin" method is illustrated in Supp. A.

Finally, our experiments yield many models that may be complementary. We implement ensembling the predictions of our best models, which Liu et al. (2019) has shown to work well for BERT. For SST and QQP tasks, we use the mode of the ensemble's predictions, and for STS task, we use the unweighted average. We utilize the best model from multi-task finetuning (BERTriad) and several top models from single-task setting (BERTrilogy), employing a strategy where models are added to each task's ensemble only if they improve or maintain the dev set performance.

[1]`https://www.kaggle.com/datasets/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews/data`

[2]`https://nlp.stanford.edu/projects/snli/`

5 Experiments

5.1 Data

Table 1: **Tasks and corresponding datasets.** "Cl." stands for classification and "Reg." for regression. The number in the brackets denotes the number of classes for the classification or the range for regression, respectively.

Task	Type	Dataset	Examples
Sentiment Analysis	Cl. (5)	Stanford Sentiment Treebank (SST-5)	$11,855$
Paraphrase Detection	Cl. (2)	Quora question pairs dataset (QQP)	$400,000$
Semantic Textual Similarity	Reg. (0-5)	SemEval STS Dataset (STS)	$8,628$

An brief overview of our downstream tasks and corresponding datasets is shown in Table 1.

For the sentiment analysis task, the Stanford Sentiment Treebank Socher et al. (2013) consists of 11,855 single sentences from movie reviews extracted from movie reviews. The dataset was parsed with the Stanford parser[3] and includes a total of 215,154 unique phrases from those parse trees, annotated by 3 human judges. Each phrase has a label of negative, somewhat negative, neutral, somewhat positive, or positive. For the paraphrase detection task, the Quora dataset Fernando and Stevenson (2008) consists of 400,000 question pairs with binary labels indicating whether particular instances are paraphrases of one another. For the semantic textual similarity task, the SemEval STS Benchmark dataset Agirre et al. (2013) consists of 8,628 different sentence pairs of varying similarity on a scale from 0 (unrelated) to 5 (equivalent meaning).

5.2 Evaluation method

Following the default project handout, we use the accuracy as the evaluation metric for both the sentiment analysis and paraphrase detection tasks. For semantic textual similarity task, we calculate the Pearson correlation of the true similarity values against the predicted similarity values in Agirre et al. (2013) as the evaluation metric.

5.3 Experimental details

All experiments are implemented in Pytorch v1.13.0 and conducted on a workstation with 1 Nvidia 3090 GPU. We run 10 epochs with a finetune learning rate of $1e-5$ and a hidden-layer dropout probability of 0.3 for all three tasks. We set our batch size to be 16. The model is optimized by AdamW algorithm with weight decay $1e-6$. We use Jiang et al. (2020)'s SMART loss with symmetric KL-divergence for the classification tasks SST and QQP, and squared error for STS regression. Using their recommended default 1 sampling step, $\epsilon = 1e-6, \sigma = 1e-5, \eta = 1e-3, p = \infty$, we experimented with λ_S as showing in Figure 4.

5.4 Results

Head Architectures. We conducted thorough experiments to find the optimal head architecture for each downstream task. As illustrated in Figure 2 (a), for the SST task, we observed that employing a Multi-Layer Perceptron as classifier yields the most favorable results, reaching an accuracy of 53.3%. For QQP task, we can see in Figure 2 (b) that using a linear layer with ReLU as activation achieves 89.3% accuracy, while for STS task, using a linear layer with Sigmoid as its activation reaches Pearson correlation of 86.5% (Figure 2 (c)). These three optimal head architectures are used in the multi-task setting.

Further Pre-training. As demonstrated in Figure 3, the overall performance improves a lot no matter which head architectures we use for downstream tasks. Notably, in SST task, the dev accuracy increases from 52.2% to 54.3%, and in STS task, the dev corr increases from 86.4% to 88.4%.

[3]`https://nlp.stanford.edu/software/lex-parser.shtml`

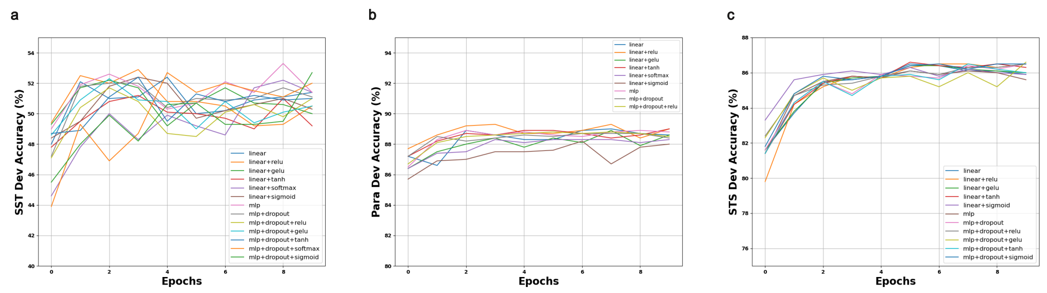

Figure 2: **The dev accuracy curves of (a) SST head architectures, (b) QQP head architectures and (c) STS head architectures.**

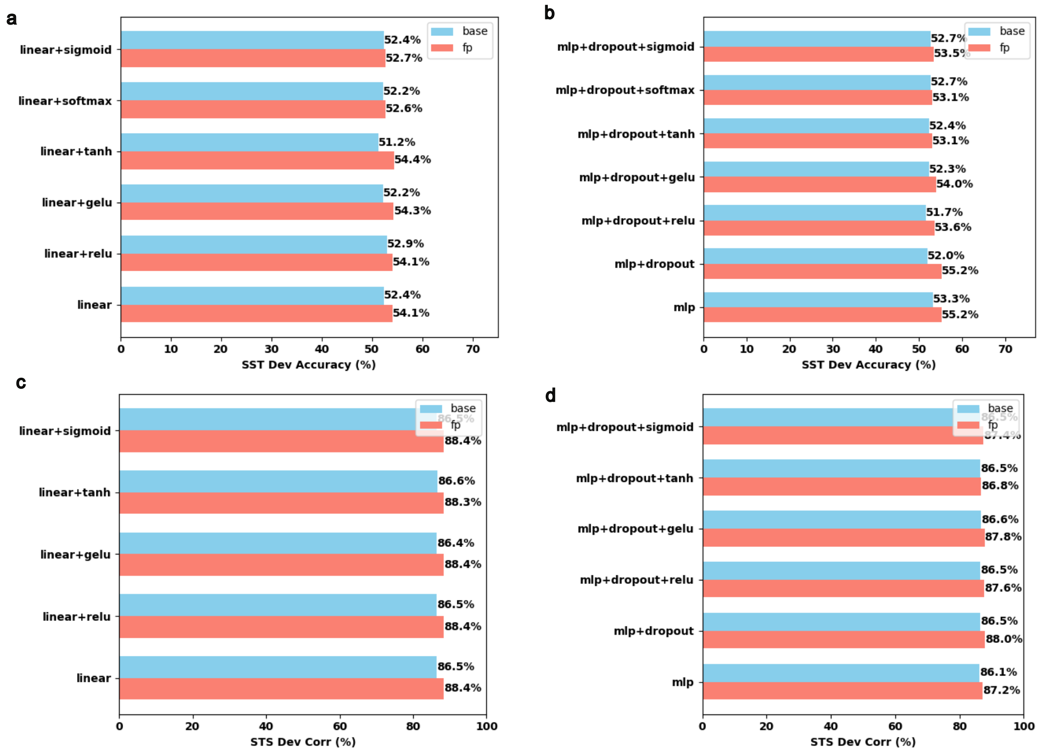

Figure 3: **Effectiveness of further pretraining.** Embedding obtainded by further pretraining tend to have better performance after task-specific finetuning.

SMART loss. As depicted in Figure 4, we observe that setting $\lambda_S = 5$ yields favorable results on the SST task, achieving an accuracy of 54.1%, an improvement from 52.0% without the SMART loss. Additionally, we note that higher weights achieve best dev accuracy in later epochs. Specifically, for $\lambda_S > 10$, the model is seen to reach its peak accuracy subsequent to 10 epochs. For instance, setting $\lambda_S = 100$ allows the model to achieve a dev accuracy of 53.2% by the 24th epoch.

Ensembling. In our methodology, we strategically leverage a combination of the most effective model emerging from multi-task finetuning, which we refer to as BERTriad, along with a selection of 4 to 6 top-performing models from a single-task setting, collectively termed BERTrilogy.On the development (dev) dataset, our ensemble method attained an impressive average accuracy of **80.2%**, while on the test dataset, we average test accuracy of **79.6%** in Table 3.

Table 2: **Dev set results of the BERTrilogy and BERTriad.**

Method		SST-5 (dev)	QQP (dev)	STS-B (dev)	Avg. (dev)
BERTrilogy	-	53.3	88.9	86.6	76.3
	SMART	54.3	89.1	87.8	77.1
	SimCSE	53.5	88.5	87.0	76.3
	SMART & SimCSE	53.6	88.7	86.8	76.4
BERTrilogy with further pre-training	-	55.0	89.2	88.3	77.5
	SMART	<u>55.2</u>	<u>89.3</u>	<u>88.4</u>	<u>77.6</u>
BERTriad	cycle-round-robin	50.0	88.4	86.4	74.9
	shuffle-round-robin	48.7	89.2	85.5	74.5
	smart-round-robin	49.9	88.8	86.7	75.1
BERTriad with further pre-training	cycle-round-robin	51.2	88.2	86.2	75.2
	shuffle-round-robin	52.4	88.5	86.9	75.9
	smart-round-robin	53.1	88.4	87.1	76.2
Ensemble	5-7 models	**56.0**	**90.0**	**89.2**	**80.2**

Table 3: **Test set results for the original and ensembled models.**

Method	SST-5 (test)	QQP (test)	STS-B (test)	Avg. (test)
Best Results before Ensembling	53.2	89.1	88.3	78.7
Ensemble	**54.3**	**90.0**	**89.1**	**79.6**

6 Analysis

We explored the enhancement of BERT model's performance through various techniques, namely SimCSE, further pre-training, SMART loss, and different multitasking approaches, culminating in an ensemble strategy. Our findings suggest significant improvements across multiple NLP tasks, demonstrating the efficacy of our proposed methodologies.

SimCSE & Further Pre-training Our results indicate that both SimCSE and further pre-training contribute substantially to the model's performance improvements. SimCSE enhances the models's ability to generate more uniform and aligned sentence embeddings. This improvement is particularly evident in the Semantic Textual Similarity Task, where the nature of the task directly benefits from accurate pair predictions.

Further pre-training on task-specific corpora helps tailor the BERT model to understand the nuances of different domains, such as movie reviews for Sentiment Analysis and question pairs for Paraphrase Detection and Semantic Textual Similarity. The significant performance increases observed in these tasks after further pre-training validate the value of customizing the pre-training phase to align closely with the downstream tasks.

SMART Loss & Multitask Learning Approaches The implementation of SMART loss aimed at mitigating the overfitting problem often encountered in fine-tuning pre-trained models on specific downstream tasks. Our experiments showed that incorporating SMART loss leads to more robust models that could generalize better to unseen data, as evidenced by the performance gains in Sentiment Analysis and Paraphrase Detection tasks. Empirically, using SMART loss weight **5** yields the best result. This is reasonable since if we take small weight, the SMART loss would basically make no contributions to the loss function. On the other hand, if we take to much larger weight, the SMART loss would take the dominance, thus prevent our model from learning the downstream task.

Our multitask learning approach, particularly the novel "smart-round-robin" routine, effectively managed the challenge of conflicting gradients in multitask learning. By facilitating gradient surgery within this framework, we observed improved performance and reduced overfitting, highlighting the potential of multitask learning to enhance model robustness and efficiency across several tasks simultaneously.

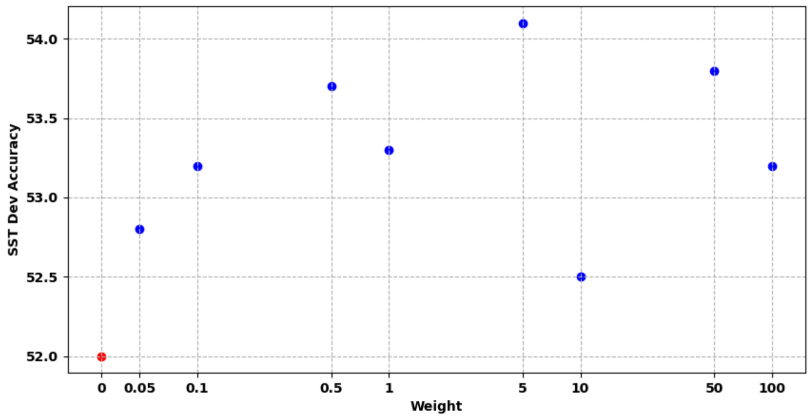

Figure 4: **Effect of SMART loss weight on SST dev accuracy.**

Ensemble Strategy The ensemble strategy, combining the strengths of individual models from both single-task and multitask settings, emerged as a powerful method to leverage diverse perspectives and strengths. This approach led to notable performance gains across all tasks, as shown in Table 2 and 3. An interesting finding is that ensembling a bunch of single task models and one multitask model could yield better performance on individual tasks than using the best single task models alone. This outcome suggests a complementary relationship between the deep, task-specific insights learned by single task models and the broad, generalizable understanding developed by the multitask model.

7 Conclusion

Our study presents a comprehensive examination and improvement of BERT models for NLP tasks, implementing and evaluating techniques such as SimCSE, further pre-training, SMART loss, and innovative multitask learning strategies. **We achieve the best dev and test accuracy on the leaderboard.** Our results underscore the importance of tailored pre-training, regularization techniques, and thoughtful integration of multitask learning to enhance model performance and generalization. The ensemble approach, combining the strengths of individual models, emerged as a particularly powerful strategy, leading to notable performance gains across different tasks. This reinforces the idea that while single models can be effective, the synergy of multiple models, each bringing unique perspectives and strengths, can lead to superior performance.

However, our work is not without limitations. The specific configurations and techniques used were tailored to our selected tasks and datasets, so further research is needed to assess their generalizability to other contexts.

For future work, we aim to explore more efficient and scalable techniques for model training and ensembling. Investigating the transferability of our approaches to other languages and more diverse sets of tasks will also be a priority. Furthermore, delving deeper into the interpretability of the ensemble models could provide valuable insights into their decision-making processes and help identify areas for further improvement. By pushing the boundaries of what's possible with BERT and NLP models, we hope to contribute to the development of more nuanced, robust, and effective tools for language processing, ultimately advancing the field and its applications.

References

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*, pages 32–43.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A simple but tough-to-beat baseline for sentence embeddings. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Rich Caruana. 1997. Multitask learning. *Machine learning*, 28:41–75.

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

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. 2018. Supervised learning of universal sentence representations from natural language inference data.

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

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? Comparing the geometry of BERT, ELMo, and GPT-2 embeddings. 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 55–65, Hong Kong, China. Association for Computational Linguistics.

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*, pages 45–52.

Jun Gao, Di He, Xu Tan, Tao Qin, Liwei Wang, and Tie-Yan Liu. 2019. Representation degeneration problem in training natural language generation models. *arXiv preprint arXiv:1907.12009*.

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

Ty Henderson and Qing Liu. 2017. Efficient design and analysis for a selective choice process. *Journal of Marketing Research*, 54(3):430–446.

Felix Hill, Kyunghyun Cho, and Anna Korhonen. 2016. Learning distributed representations of sentences from unlabelled data. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1367–1377, San Diego, California. 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.

Ryan Kiros, Yukun Zhu, Russ R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. *Advances in neural information processing systems*, 28.

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. On the sentence embeddings from pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9119–9130, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

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. *Advances in neural information processing systems*, 30.

Tongzhou Wang and Phillip Isola. 2020. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International conference on machine learning*, pages 9929–9939. PMLR.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. 2020. Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems*, 33:5824–5836.

A Algorithm of Smart-Round-Robin

Algorithm 1 Smart-Round-Robin with Gradient Surgery

```
1:  $maxIterations \leftarrow len(largestDataloader)$
2:  Initialize $optimizer$ with PCGrad and Adam parameters
3:  for $epoch = 1$ to $epochs$ do
4:      $sst\_iter, ... \leftarrow iterator(sst\_train\_dataloader), ...$
5:      $sst\_batches\_processed, ... \leftarrow 0$
6:      for $i = 1$ to $maxIterations$ do
7:          $losses \leftarrow []$
8:          $number \leftarrow random(0, 1)$
9:          if $len(sst\_train\_dataloader) - sst\_batches\_processed \geq max\_iterations - i$ or
    $number \leq \frac{len(sst\_train\_dataloader)}{max\_iterations}$ then
10:             $sst\_batch \leftarrow next(sst\_iter)$
11:             $sst\_batches\_processed \leftarrow sst\_batches\_processed + 1$
12:             $optimizer.zero\_grad()$
13:             $loss\_sst \leftarrow model(sst\_batch)$
14:             Append $loss\_sst$ to $losses$
15:         end if
16:         ...
17:         $optimizer.pc\_backward(losses)$
18:         $optimizer.step()$
19:     end for
20: end for
```

B SMART: Adversarial Loss Calculation

Algorithm 2 SMART: Adversarial Loss Calculation

```
1: **Data**: $\hat{f}$: model, $\{x_i\}_1^Q$: model embeddings, $\{z_j\}_1^B$: batch inputs, $\sigma^2$: variance of noise,
   $\eta = 10^{-3}$: learning rate for adversarial update, $\epsilon = 10^{-5}$: max perturbation.
2:
3: **begin**
4:     $\hat{f}$ set to eval mode
5:     $y_j \leftarrow \hat{f}(z_j, x)$
6:     for $x_i \in \mathcal{Q}$ do
7:         $\hat{x}_i \leftarrow x_i + v_i$ with $v_i \sim \mathcal{N}(0, \sigma^2 I)$
8:     end
9:     for $y_j \in \mathcal{B}$ do
10:        $\tilde{h}_j \leftarrow \nabla_{\tilde{x}} l_s(y, \hat{f}(z_j, x))$
11:    end
12:    $\tilde{g}_i \leftarrow \left(\frac{1}{|B|}\sum_i \tilde{h}_j\right)\left(\|\frac{1}{|B|}\sum_i \tilde{h}_j\|_\infty\right)^{-1}$
13:    for $x_i \in \mathcal{Q}$ do
14:        $\tilde{x}_i \leftarrow \mathcal{P}_{\|\tilde{x}_i + \eta\tilde{g}_i - x_i\| \leq \epsilon}(\tilde{x}_i + \eta\tilde{g}_i)$
15:    end
16:    $y_j^{adv} \leftarrow \hat{f}(z_j, \tilde{x})$
17:    $\mathcal{L}_{adv,classification} \leftarrow \frac{1}{|\mathcal{B}|}\sum_{j\in\mathcal{B}} \mathcal{D}_{KL}(y_j \| y_j^{adv}) + \mathcal{D}_{KL}(y_j^{adv} \| y_j)$
18:    $\mathcal{L}_{adv,regression} \leftarrow \frac{1}{|\mathcal{B}|}\sum_{j\in\mathcal{B}} \|y_j - y_j^{adv}\|^2$
19:    $\hat{f}$ set to train mode
20: **end**
```