Multi-task Learning with minBERT

Stanford CS224N Default Project

Praneet Bhoj
Department of Computer Science
Stanford University
praneet@stanford.edu

Abstract

Though the BERT architecture has had much success in understanding and producing natural language sequences, eliciting its generalist capabilities on multiple downstream tasks simultaneously is not as simple as appending task-specific adapter heads on top of the core BERT model in a disconnected manner. Based on this observation, our work presented in this paper seeks to apply various training and architecture approaches to a pre-trained minBERT model in order to develop a single model that demonstrates improved proficiency on 3 downstream tasks: sentiment analysis, paraphrase detection, and semantic textual similarity. We find that using a shared multi-task loss coupled with cosine similarity scoring (for semantic textual similarity), gradient surgery, and smoothness-inducing adversarial regularization enables effective updates of shared minBERT embedding parameters to yield improved performance across all three tasks relative to a baseline model with task-specific adapters trained in a disconnected manner. However, we also observe that our improved model shows an over-reliance on a small set of adjective tokens for sentiment analysis and severe overfitting for semantic textual similarity, which highlight potential avenues for future exploration.

1 Key Information to include

- Mentor: Olivia Lee
- External Collaborators (if you have any): N/A
- Sharing project: N/A

2 Introduction

Since the introduction of the Transformer architecture [1], artificial intelligence for natural language processing has seen tremendous increases in capabilities across a wide variety of tasks. Indeed, Transformer-based models like BERT [2] have shown great results in processing natural language input and generating coherent, human-like natural language output. However, while generalist foundation models such as BERT can perform a large number of tasks to some degree, their performance on more specialized downstream tasks may not be adequate for practical use. Therefore, to develop more useful models for a smaller set of domain-specific downstream tasks, it is necessary to fine-tune our foundation models on domain-specific data while ensuring that it is done with a multi-task learning approach in order to maintain the model's generalizability among the downstream tasks of interest.

While multi-task learning can be approached by having task-specific layers appended to the base foundation model architecture and trained in a disconnected manner, this approach does not update the parameters of the base model and thus does not leverage any shared embedding structure between the different tasks. To take advantage of this potential shared information in the embedding space, multi-task learning can instead propagate parameter updates all the way from the task-specific layers through the base foundation model. However, this naturally presents issues due to the fact that the

task-specific gradients for the shared parameters may point in very different directions, thus causing conflicting updates to those shared parameters.

In this paper, we explore the concept of multi-task learning using a simplified version of the BERT model that will be referred to as minBERT. In particular, we look to use multi-task learning to improve the performance of a single minBERT model on the following three different downstream tasks: sentiment analysis of natural language sequences, paraphrase detection between two natural language sequences, and semantic textual similarity between two natural language sequences. This work experiments with an approach to the issue of conflicting gradients in multi-task learning called gradient surgery [3], and we augment it with regularization and cosine similarity techniques to demonstrate improved proficiency across all 3 downstream tasks using the single minBERT model.

3 Related Work

Given the tremendous advances in language models in recent years, there has been a large amount of interest and work in multi-task learning. Large language models including BERT have been shown to indeed have significant multi-task learning capabilities provided sufficient amounts of pretraining data [4]. More recent work has also supported the idea that multi-task learning during pre-training can improve model performance on desired downstream tasks [5]. However, these approaches require a large amount of data to be successful, and the data inefficiency introduced by multi-task pre-training makes training the large language models even more difficult. So the multi-task pre-training option is not always viable for all cases in which models need to be developed for domain-specific downstream tasks. Therefore, as an alternative to multi-task pre-training, previous work has also looked at fine-tuning on multiple tasks simultaneously with updates to the shared parameters to allow for the model to learn shared task input representations, but properties such as the aforementioned trouble of conflicting shared parameter updates during training make the optimization procedure quite difficult and thus yields poor overall results [6, 7].

To address this problem of conflicting gradients during parameter optimization, previous work has developed strategies for mitigating gradient conflicts by adjusting the calculated gradients for each task such that they are better aligned. In particular, the technique called gradient surgery involves projecting a task's gradient onto the normal plane of any conflicting gradient from any other task [3]. By aligning task-specific gradients in this way, the shared parameters are able to be updated more smoothly and thus the network overall can learn more efficiently. Based on this strategy, existing work has utilized gradient surgery to improve a news recommendation system built on a BERT architecture [8]. However, that work formulated the problem as a single main task combined with two lesser-weighted auxiliary tasks whose purpose was only to improve the performance on the main task. Therefore, the work presented in this paper looks to further analyze the impact of gradient surgery by applying it in a context of learning three equally-weighted tasks (sentiment analysis, paraphrase detection, and semantic textual similarity) with a minBERT model. Additionally, we do not use gradient surgery in isolation as a single enhancement, but rather combine it with a regularization strategy and cosine similarity calculation to explore their combined effects on multi-task performance.

4 Approach

The core of the approach presented in this paper is focused on the minBERT architecture, which provides a strong foundation for converting input sequences of natural language into context-aware vector embeddings. The model achieves this by first padding the input sequence to length 512, tokenizing the padded input based on a predetermined set of 30000 tokens, and converting the input sequence tokens into IDs using their indices in the token set. The tokenized input sequence is then passed through a trainable embedding layer to get a 768-dimensional embedding for each input token, followed by the addition of a learnable positional embedding value that is calculated for each of the 512 input positions. Finally, the token embeddings are adjusted to their contexts using 12 encoder Transformer layers.

Each head i of the n-headed multi-head attention mechanism used in the minBERT Transformer encoder layers iterates over the T input sequence token embeddings $\mathbf{h}_j$ and produces attention scores based on the d-dimensional embeddings using the dot-product attention as seen in Equation 1 originally presented in [1].

$$\text{Attention}_i(\mathbf{h}_j) = \sum_{t=1}^{T} \text{softmax}\left(\frac{W_i^q \mathbf{h}_j \cdot W_i^k \mathbf{h}_t}{\sqrt{d/n}}\right) W_i^v \mathbf{h}_t \tag{1}$$

Following the multi-head attention, add-norm, and feed-foward components of the Transformer encoder layers, the minBERT model outputs the result of the final Transformer layer along with a pooled representation of the special `[CLS]` token that is prepended to each input sequence. To transform the embeddings into the desired outputs for the three downstream tasks (sentiment analysis, paraphrase detection, and semantic textual similarity), three separate task-specific heads are added on top of the model, where each head is composed of 3 linear layers with GELU [9] activation in between the layers. In the case of the paraphrase detection and semantic textual similarity tasks, since there are two input sentences for these tasks, the embeddings for the input sentences resulting from the minBERT embedding process are concatenated before being put through the task head. The baseline architecture for this work is thus defined as the frozen pre-trained minBERT model with independently fine-tuned task-specific heads (i.e. each head is fine-tuned separately with no updates to the shared minBERT model weights), and is presented in Figure 1.

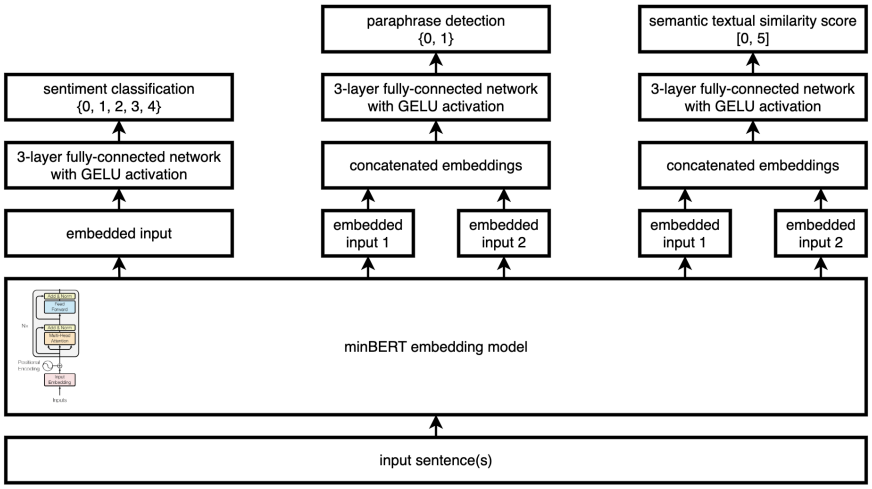

Figure 1: Baseline model structure used for our study of multi-task learning. The minBERT model is built on the Transformer encoder architecture introduced in [1] as seen within the minBERT component in the figure.

To improve upon the baseline architecture, we implemented and experimented with 3 architecture enhancements based on previous work. The usage and results of these enhancements will be described in Section 5, but the details of each enhancement are presented here.

The first enhancement used in this work is inspired by work done in [10], which shows strong results on semantic textual similarity results by using siamese BERT embedding models with cosine similarity comparisons between input embeddings. Based on this, rather than using fully-connected linear layers for the semantic textual similarity task head, we explore the direct use of cosine similarity on the two input sentence embeddings calculated using Equation 2 and then scaled to match the desired output range.

$$\text{cosine similarity}(\mathbf{x}_1, \mathbf{x}_2) = \frac{\mathbf{x}_1 \cdot \mathbf{x}_2}{\|\mathbf{x}_1\|_2 \|\mathbf{x}_2\|_2} \tag{2}$$

The next enhancement implemented in this work is gradient surgery, which is presented in [3]. As described previously, this technique performs gradient projections to align task-specific gradients and reduce conflicting parameter updates. For each task-specific gradient $\mathbf{g}_i$, the task-specific gradients $\mathbf{g}_j$ for the other tasks are iterated over in random order. If the cosine similarity between $\mathbf{g}_i$ and $\mathbf{g}_j$ is negative, this indicates conflicting gradients, and $\mathbf{g}_i$ is then projected onto the normal plane of $\mathbf{g}_j$ following Equation 3.

$$\mathbf{g}_i = \mathbf{g}_i - \frac{\mathbf{g}_i \cdot \mathbf{g}_j}{\|\mathbf{g}_j\|_2^2} \mathbf{g}_j \tag{3}$$

The final enhancement implemented in this work is smoothness-inducing adversarial regularization as presented in [11]. Using this regularization strategy yielded state-of-the-art results on a number of GLUE benchmark tasks. For a model $f(\cdot;\theta)$ and n task data input embeddings $\mathbf{x}_i$ with labels y_i, the regularized objective is given in Equation 4. The overall loss function is defined in Equation 5 with $\ell(\cdot,\cdot)$ being the task-specific loss function used for a given sample. For the regularization part, a perturbation sampled from $\mathcal{N}(0, 1 \times 10^{-5})$ is added to the original input embedding to produce a perturbed embedding $\tilde{\mathbf{x}}_i$. This is then used to calculate the smoothness-inducing adversarial regularizer defined in Equation 6 that is scaled by a strength parameter λ and added to the objective. Within the regularizer, the function $\ell_s(\cdot,\cdot)$ is defined as the symmetrized KL-divergence (Equation 7) for the sentiment analysis and paraphrase detection classification tasks, and squared error (Equation 8) for the semantic textual similarity regression task. Note that while [11] tries to find the perturbed input $\tilde{\mathbf{x}}_i$ within the neighborhood of $\mathbf{x}_i$ that maximizes ℓ_s, they also suggest that the maximization can be approximated well enough with just one random perturbation, and thus that is the approach used in this paper.

$$\mathcal{F}(\theta) = \mathcal{L}(\theta) + \lambda \mathcal{R}_s(\theta) \tag{4}$$

$$\mathcal{L}(\theta) = \frac{1}{n} \sum_{i=1}^{n} \ell(f(\mathbf{x}_i;\theta), y_i) \tag{5}$$

$$\mathcal{R}_s(\theta) = \frac{1}{n} \sum_{i=1}^{n} \ell_s(f(\tilde{\mathbf{x}}_i;\theta), f(\mathbf{x}_i;\theta)) \tag{6}$$

$$\ell_s(P, Q) = \mathcal{D}_{\text{KL}}(P||Q) + \mathcal{D}_{\text{KL}}(Q||P) \tag{7}$$

$$\ell_s(p, q) = (p - q)^2 \tag{8}$$

Across all of these enhancements to the model architecture, the task-specific loss functions used in this work are as follows: the sentiment analysis task head uses cross entropy loss, the paraphrase detection task head uses binary cross entropy loss, and the semantic textual similarity task head uses mean squared error loss. Additionally, during training, the appropriate model parameters are updated using an AdamW optimizer [12].

5 Experiments

In this section, we describe the experiments performed in our work and present their results.

5.1 Data

All of the experiments conducted in this work utilized the following datasets: 11855 sentences labeled as "negative", "somewhat negative", "neutral", "somewhat positive", or "positive" from the Stanford Sentiment Treebank [13] for the sentiment analysis task, 404298 sentence pairs labeled as paraphrases (or not) from Quora [14] for the paraphrase detection task, and 8628 sentence pairs labeled with similarity scores (0 to 5) from SemEval [15] for the semantic textual similarity task.

5.2 Evaluation method

The sentiment analysis and paraphrase detection tasks have data with discrete labels. The experimental models are thus evaluated on these tasks using simple accuracy measurements (i.e. ratio of evaluation samples with correctly predicted labels to the total number of evaluation samples). However, the semantic textual similarity task has data with labels that are continuous values (similarity scores between 0 and 5), and thus prediction accuracy is not a suitable method of evaluation for this task. Therefore, the experimental models are instead evaluated on this task using the Pearson correlation between the predicted and ground truth scores as described in [15]. Finally, an overall score is calculated from the individual task metrics with score $=$ ((sentiment analysis accuracy) $+$ ((1 $+$ (semantic textual similarity correlation))/2) $+$ (paraphrase detection accuracy))/3.

5.3 Experimental details

Having defined the datasets and evaluation metrics, we now present the details of the various experiments performed in our study of multitask learning with minBERT. The experiment descriptions are provided here, and the results of each experiment are provided in Section 5.4.

The initial experiment we ran in our work deals with variations in the fine-tuning sequence for the downstream tasks. For the baseline predictions, the task-specific heads are each fine-tuned for 10 epochs on top of a frozen minBERT model. However, given the relationships between the tasks, it seems likely that there may be some shared weights or structure in the embedding process such that tuning the minBERT model along with the task heads can yield a performance improvement across the board. Based on this hypothesis, the task heads and minBERT model were fine-tuned by doing a full pass through each of the task training datasets and making parameter updates based on the task-specific loss during each epoch. The order in which these tasks were processed during each epoch was varied for the experiment, giving 6 different model configurations (all possible orderings of the 3 tasks). As a 7th and final model configuration for this experiment, we looked at an alternate approach to the independent fine-tuning, which is to perform multi-task learning through summed losses as seen in [8]. In this configuration, each training batch contained equal amounts of data for all 3 tasks, and after calculating the appropriate losses for data corresponding to each task, the losses are added together and full model gradients are taken with respect to this summed loss for parameter updates. Rather than training for a full 10 epochs on all 7 of these configurations, the models were trained for 1 epoch each to get a sense of which configuration had the most promising results. The results, presented in Table 1, show that the summed loss configuration produced the best early results from our brief fine-tuning procedure, and it is thus selected as the configuration for further experimentation and iteration in the remainder of our work.

Following our training configuration experiment, we looked at the impact of replacing the fully-connected linear layers of our semantic textual similarity task head with a direct cosine similarity score. We believed that this may be a more suitable approach to the semantic textual similarity task given that we are looking to quantify the similarity between two input embedding vectors as our underlying objective. This approach follows what was described in Section 4, where the two input sentences are each embedded using the same minBERT embedding network weights, and then their embedding vectors are directly passed through a cosine similarity function whose output is scaled to the required 0 to 5 output range. The summed loss model configurations with the fully-connected STS task head and the cosine similarity STS task head were each trained for 10 epochs, with the results in Table 2 showing that the model using cosine similarity performs slightly better and is thus used for the next round of experimentation.

Building on the initial cosine similarity enhancement, we then experimented with gradient surgery to improve overall multi-task performance. Given that multi-task learning can often produce conflicting gradient directions as described previously, we implement gradient surgery in our network to produce more aligned parameter updates for smoother learning. The implementation of gradient surgery follows what is presented in Section 4, where gradients are projected onto the normal plane of conflicting gradients. The model utilizing the gradient surgery implementation was trained for 10 epochs. The results shown in Table 3 highlight a small performance increase from gradient surgery, so we carry this implementation forward for our next experiment.

Following the implementation of cosine similarity scoring and gradient surgery on our summed loss multi-task model, we observed signs of overfitting in the form of diverging training and validation accuracies/correlations for our three tasks. Therefore next experiment we performed explores the utility of regularization in our multi-task learning process. To mitigate the effects of overfitting, we implemented smoothness-inducing adversarial regularization as described in Section 4, which includes an addition to the standard loss using a regularization term based on perturbed input embeddings. As with the previous experiments, this model with the regularization implementation was trained for 10 epochs. The results for this experiment are given in Table 4, and show that the inclusion of regularization provides another small boost in overall performance for our model. It is therefore kept in our model implementation for our final experiment.

The final experiment we conducted in our study is a hyperparameter sweep to identify the optimal hyperparameter configuration for our enhanced multi-task model. Inspired in part by the work in [11], our hyperparameter sweep includes learning rates lr $\in \{1 \times 10^{-3}, 1 \times 10^{-4}, 1 \times 10^{-5}\}$, regularization strength $\lambda \in \{1, 3, 5, 7\}$, and intermediate layer dropout rates d $\in \{0.1, 0.3, 0.5\}$ for

a total of 36 hyperparameter configurations. We trained our model (which now includes summed task loss, cosine similarity, gradient surgery, and smoothness-inducing adversarial regularization) using these hyperparameter configurations for 2 epochs each. Using the validation results from the experiment, we found that the best-performing hyperparameter configuration uses lr $= 1 \times 10^{-4}$, $\lambda = 5$, and d $= 0.1$. This configuration is then trained for a full 10 epochs to produce our final multi-task model.

5.4 Results

The developmental results for the experiments detailed in Section 5.3 are presented here, along with the actual test results for our final model configuration for the sentiment analysis (SA), paraphrase detection (PD), and semantic textual similarity (STS) tasks.

Model Configuration	SA Dev Acc	PD Dev Acc	STS Dev Corr	Overall Score
Baseline	0.387	0.678	0.253	0.564
1 (SA, PD, STS)	0.264	0.623	0.338	0.518
2 (SA, STS, PD)	0.190	0.787	0.124	0.513
3 (PD, SA, STS)	0.456	0.765	0.342	0.631
4 (PD, STS, SA)	0.487	0.740	0.231	0.614
5 (STS, SA, PD)	0.222	0.787	0.120	0.523
6 (STS, PD, SA)	0.505	0.778	0.169	0.623
Multi-task Summed Loss	0.480	0.730	0.391	0.635

Table 1: Validation results for the initial model configuration experiment. For the sequential fine-tuning configurations 1-6, the order in which the tasks are fine-tuned on during each epoch is indicated in parentheses.

Table 1 presents the results from our initial model configuration experiment. We observe that the summed loss model shows more promising results than any of the sequentially trained models. Though this is somewhat surprising, it may be attributed to the fact that a model can theoretically learn better/more efficiently if it gets signals from all tasks it is optimizing at the same time. It could also potentially be attributed to the fact that sequential approaches for multi-task learning can introduce issues of forgetting, where a model trained on data for a particular task forgets how to perform tasks it was previously trained on.

Model Configuration	SA Dev Acc	PD Dev Acc	STS Dev Corr	Overall Score
without cosine similarity	0.482	0.734	0.383	0.636
with cosine similarity	0.491	0.712	0.402	0.638

Table 2: Validation results for the cosine similarity experiment.

Table 2 shows the results from our embedding vector cosine similarity experiment. Though the overall performance increase from the addition of cosine similarity is small, the metric of note is the semantic textual similarity correlation. The improved correlation metric on this task is expected given that the directional similarity between two input sentence embedding vectors should indeed provide a strong indicator of semantic similarity. Worsening performance on this task using cosine similarity would have suggested that our learned embeddings are not adequately capturing semantic meaning as is desired.

Model Configuration	SA Dev Acc	PD Dev Acc	STS Dev Corr	Overall Score
without gradient surgery	0.491	0.712	0.402	0.638
with gradient surgery	0.482	0.742	0.416	0.644

Table 3: Validation results for the gradient surgery experiment.

Table 3 contains the results from our gradient surgery experiment. We do see an overall multi-task performance increase with the introduction of gradient surgery, however the magnitude of this increase is lower than anticipated. This suggests that, despite the differences in the three tasks of interest, gradient conflicts between these tasks are not so prevalent to the point that they are causing strongly

Model Configuration	SA Dev Acc	PD Dev Acc	STS Dev Corr	Overall Score
without regularization	0.482	0.742	0.416	0.644
with regularization	0.489	0.727	0.448	0.647

Table 4: Validation results for the smoothness-inducing adversarial regularization experiment.

opposing parameter updates, and thus the gradient alignment only offers a smaller-than-expected performance gain.

Table 4 gives the results from our smoothness-inducing adversarial regularization experiment. Though a performance increase does result from implementing this regularization technique, it is quite notable that the regularization only slightly mitigates the overfitting issue. The training/validation curves shown in Figure 2 still show evidence of severe overfitting (especially in sentiment analysis and semantic textual similarity tasks), highlighting the fact that further work on this topic will need to focus on additional analysis of the input embedding perturbations used in this regularization technique, or focus on new regularization techniques altogether.

Table 6 has the results from our final hyperparameter sweep over 36 combinations of learning rate, regularization strength, and dropout rate. Though hyperparameters are often empirically determined, it is notable that the optimal regularization strength for this configuration is not the max value in light of the overfitting issues discussed above. It may be that the presence of dropout in the intermediate layers reduces the need for smoothness-inducing adversarial regularization to yield increased performance, but a more definitive answer may arise from even more fine-grained hyperparameter sweeps in future iterations of this work.

Task	Test Metric
Sentiment Analysis	0.481 (accuracy)
Paraphrase Detection	0.741 (accuracy)
Semantic Textual Similarity	0.409 (correlation)
Overall Score	0.642 (score)

Table 5: Test results for the final model. The selected final model uses multi-task summed loss and includes input embedding cosine similarity for semantic textual similarity, gradient surgery, and smoothness-inducing adversarial regularization. The hyperparameters selected for training this model are learning rate 1×10^{-4}, regularization strength 5, and dropout 0.1, and the model is trained with this configuration for 10 epochs.

After conducting these experiments and performing the full 10-epoch training procedure on the selected model configuration, we present the test data results for the final model in Table 5. Though sentiment analysis is quite a difficult and subjective task, the semantic textual similarity metric is relatively low compared to what may be anticipated or desired on that task. It seems that despite the efforts to learn cosine similarity between input embedding vectors, they are still not very representative of semantic meaning in the sentence. It may require additional fine-tuning for a larger number of epochs for this property to emerge, though a re-consideration of the initial fully-connected linear layers versus cosine similarity task head could be explored as well considering these results. However, there is still an overall performance increase relative to developmental baseline results, which suggests that the summed loss approach with gradient surgery and regularization may still hold potential for even greater performance gains through additional efforts such as fine-grained hyperparameter sweeps.

6 Analysis

The experiments we conducted as part of our exploration of multi-task learning with minBERT loosely resemble an ablation study over the various proposed model enhancements, showing the multi-task performance impact of each enhancement incrementally added on to the model. In particular, we observe a consistent performance increase when each of cosine similarity, gradient surgery, and smoothness-inducing adversarial regularization are introduced into the model, however the relatively small boost from gradient surgery is an indicator that our model is producing only a small number of conflicting gradients when training on our three tasks.

The performance of our final model can also be qualitatively assessed by looking at patterns in inputs that produce failing outputs, particularly focusing on the sentiment analysis and semantic textual similarity tasks that showed the worst results. For sentiment analysis, the model performs well in assessing input sentences with commonly positive/negative adjectives. For example, the input sentence `[a fast, funny, highly enjoyable movie]` is accurately deemed to have a positive sentiment, and we observe high accuracy on similarly structured inputs. However, this reliance on adjectives well-associated with particular sentiments also appears to lead to many of the observed incorrect predictions. For example, the input sentence `[uses sharp humor and insight into human nature to examine class conflict, adolescent yearning, the roots of friendship and sexual identity]` is predicted as only neutral, when the true classification is positive. On inputs like these, the model fails to capture the sentiment of nuanced film reviews in the absence of common positive or negative descriptors like funny, entertaining, engaging, slow, boring, tedious, etc. that are often present in the correctly-classified inputs. Despite the model's shortcomings here, it can also be noted that some of the inputs are difficult to classify due to the inherent subjectivity of sentiment analysis. For example, `[a rigorously structured and exquisitely filmed drama about a father and son connection that is a brief shooting star of love]` is reasonably predicted as a fully positive sentence (as the movie is said to be "exquisitely filmed"), but the true label is only somewhat positive. Arguments for either classification can be made here, but this is also indicative of a potential need for normalization across reviewer scores to mitigate some of the subjectivity. Furthermore, it appears that the model's focus on sentiment adjectives also impacts its performance on semantic textual similarity. For example, the input sentences `[someone is stirring noodles in water]` and `[a woman is boiling noodles in water]` are determined have a similarity score of 4.7 from the model, while the true score is only 3.2. It seems that the model has not developed a strong assessment of synonyms, and thus is overly reliant on pure token matches when scoring sentence similarity like in the previous example. This is further exemplified by the example input sentences `[two black dogs are playing on the grass]` and `[two black dogs are playing in a grassy plain]`, which are accurately given a similarity score of 4.7 (with the true score being 4.6) due to the abundance of matching tokens between the two. The model may require extended training duration to develop better synonym detection to improve its textual similarity performance beyond token matching, though it is important to note that the high training scores for semantic textual similarity show that the model has the expressive capability to learn these properties, and performance improvement on validation and test data for this task may instead be achieved through more rigorous regularization techniques.

7 Conclusion

In our work, we explore the application of minBERT to a multi-task learning problem covering three natural language processing tasks: sentiment analysis, paraphrase detection, and semantic textual similarity. We demonstrate that multi-task learning with shared loss backpropagated through the entire model shows greater potential over models trained sequentially on different tasks. Furthermore, we find that the shared loss multi-task learning procedure can be coupled with gradient surgery and regularization techniques to yield performance increases across all three tasks of interest, though the small observed gain from gradient surgery suggests that conflicting gradients may not be as prevalent as anticipated during training on these tasks.

However, despite the performance gains shown in our experimentation, there is still room for further improvement. In particular, further training may be required to allow the model to accurately analyze more nuanced inputs that don't contain well-defined sentiment descriptors (for sentiment analysis) or matching tokens (for semantic textual similarity). It is also apparent that the regularization technique implemented in our work fails to truly mitigate the issue of overfitting, and thus future work may also further analyze the regularization procedure or look into alternate approaches.

8 Ethics Statement

As is the case with many artificial intelligence developments in recent years, there are a number of societal risks or impacts that may arise from the widespread deployment of natural language processing systems like this one.

In particular, if the type of multi-task model explored in this work is deployed at scale, one risk that may arise is the fact that automated sentiment analysis has the potential to propagate biases. In this work, we look at ways to improve sentiment analysis on input token sequences without regard to the token types, and although we train our model on a relatively small dataset, performing this training on internet-scale data could cause the model to learn to associate certain people/traits/characteristics with specific sentiments. This can quickly become harmful in propagating biases through offensive sentiment associations in generated content on large-scale systems. However, there is potential mitigation for this issue in the form of data pre-processing that looks for tokens related to gender, race, etc. and randomizes them such that the model doesn't learn text sentiments based on those particular terms.

Another risk that may come up from general use of models like the one we work with stems from the lack of expressiveness of text in semantic textual similarity tasks. More specifically, transcribing text from audio of speeches/conversations will fail to capture meaning embedded in voice intonations, and thus purely token-based similarity comparisons between such inputs may suffer. For example, future video or audio recommendation systems based on spoken content relying on this type of similarity analysis may present incorrect or even offensive/damaging recommendations to users. As a possible mitigation for this risk, it may be necessary to augment transcribed text data with specialized tone-of-voice tokens from which the model can learn to parse concepts like sarcasm that arise from speech style.

References

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

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

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

[4] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[5] Da Li, Boqing Zhu, Sen Yang, Kele Xu, Ming Yi, Yukai He, and Huaimin Wang. Multi-task pre-training language model for semantic network completion. *ACM Transactions on Asian and Low-Resource Language Information Processing*, 22(11):1–20, 2023.

[6] Emilio Parisotto, Jimmy Lei Ba, and Ruslan Salakhutdinov. Actor-mimic: Deep multitask and transfer reinforcement learning. *arXiv preprint arXiv:1511.06342*, 2015.

[7] Andrei A Rusu, Sergio Gomez Colmenarejo, Caglar Gulcehre, Guillaume Desjardins, James Kirkpatrick, Razvan Pascanu, Volodymyr Mnih, Koray Kavukcuoglu, and Raia Hadsell. Policy distillation. *arXiv preprint arXiv:1511.06295*, 2015.

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

[9] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[10] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

[11] Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Tuo Zhao. Smart: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization. *arXiv preprint arXiv:1911.03437*, 2019.

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

[13] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In David Yarowsky, Timothy Baldwin, Anna Korhonen, Karen Livescu, and Steven Bethard, editors, *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA, October 2013. Association for Computational Linguistics.

[14] Samuel Fernando and Mark Stevenson. 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, 2008.

[15] Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. * 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, 2013.

A Appendix (optional)

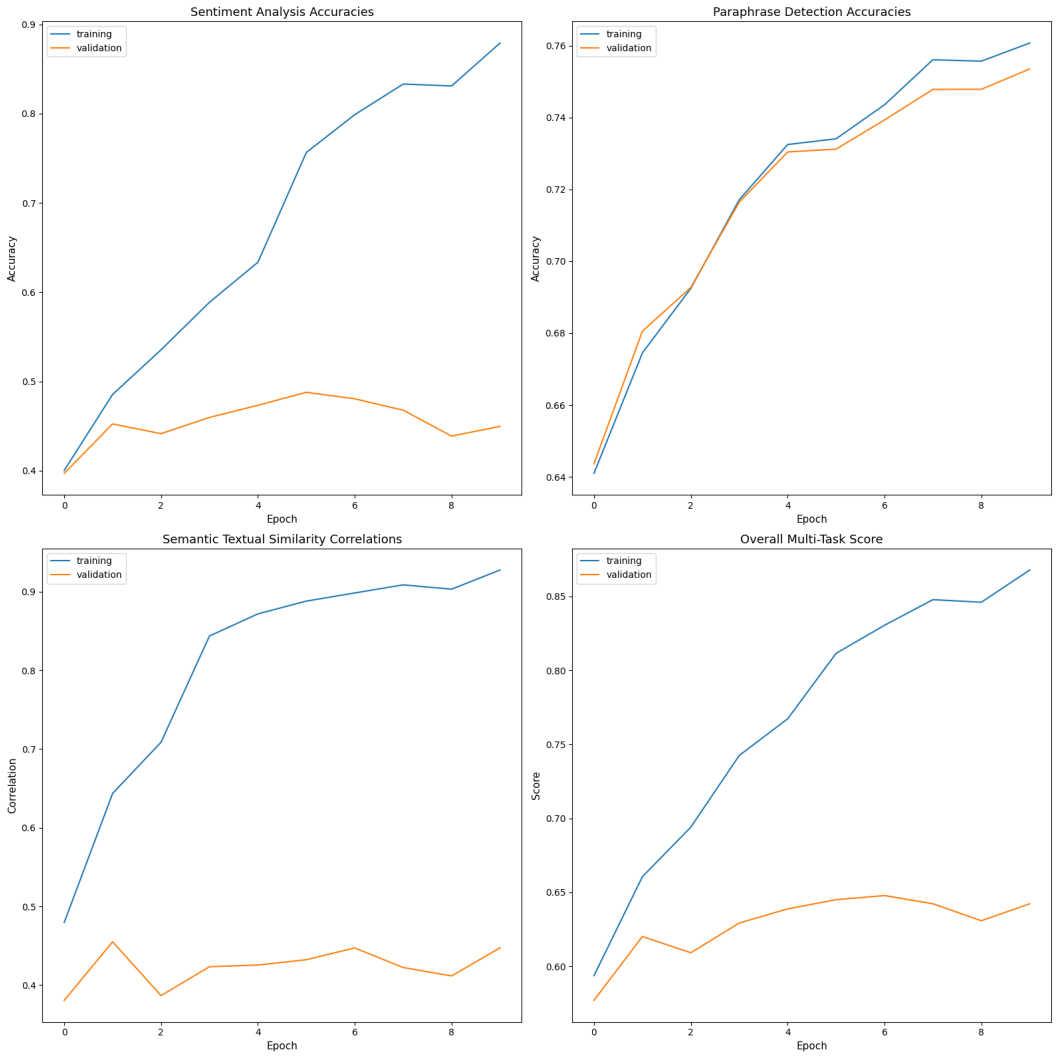

Figure 2: Training and validation accuracies, correlations, and overall scores for sentiment analysis, paraphrase detection, and semantic textual similarity over 10 epochs of training. The plot trends show signs of severe overfitting, particularly for sentiment analysis and semantic textual similarity.

Learning Rate	Reg	Dropout	SA Dev Acc	PD Dev Acc	STS Dev Corr	Overall Score
1×10^{-3}	1	0.1	0.262	0.632	NaN	NaN
1×10^{-3}	1	0.3	0.262	0.632	-0.033	0.459
1×10^{-3}	1	0.5	0.262	0.632	0.028	0.469
1×10^{-3}	3	0.1	0.262	0.632	NaN	NaN
1×10^{-3}	3	0.3	0.262	0.632	NaN	NaN
1×10^{-3}	3	0.5	0.262	0.632	NaN	NaN
1×10^{-3}	5	0.1	0.262	0.632	NaN	NaN
1×10^{-3}	5	0.3	0.262	0.632	NaN	NaN
1×10^{-3}	5	0.5	0.262	0.632	NaN	NaN
1×10^{-3}	7	0.1	0.262	0.632	-0.063	0.454
1×10^{-3}	7	0.3	0.262	0.632	NaN	NaN
1×10^{-3}	7	0.5	0.262	0.632	NaN	NaN
1×10^{-4}	1	0.1	0.422	0.648	0.316	0.576
1×10^{-4}	1	0.3	0.253	0.632	0.276	0.508
1×10^{-4}	1	0.5	0.385	0.638	0.369	0.569
1×10^{-4}	3	0.1	0.253	0.632	0.159	0.488
1×10^{-4}	3	0.3	0.362	0.642	0.315	0.554
1×10^{-4}	3	0.5	0.286	0.631	0.309	0.524
1×10^{-4}	5	0.1	0.452	0.681	0.455	0.620
1×10^{-4}	5	0.3	0.262	0.632	0.063	0.475
1×10^{-4}	5	0.5	0.262	0.632	0.063	0.475
1×10^{-4}	7	0.1	0.423	0.678	0.345	0.591
1×10^{-4}	7	0.3	0.391	0.670	0.388	0.585
1×10^{-4}	7	0.5	0.262	0.632	-0.058	0.455
1×10^{-5}	1	0.1	0.368	0.652	0.356	0.566
1×10^{-5}	1	0.3	0.402	0.632	0.313	0.564
1×10^{-5}	1	0.5	0.388	0.633	0.353	0.566
1×10^{-5}	3	0.1	0.379	0.636	0.340	0.561
1×10^{-5}	3	0.3	0.402	0.632	0.377	0.574
1×10^{-5}	3	0.5	0.388	0.632	0.383	0.570
1×10^{-5}	5	0.1	0.363	0.632	0.282	0.545
1×10^{-5}	5	0.3	0.387	0.633	0.374	0.569
1×10^{-5}	5	0.5	0.362	0.652	0.333	0.560
1×10^{-5}	7	0.1	0.365	0.644	0.332	0.558
1×10^{-5}	7	0.3	0.369	0.654	0.394	0.573
1×10^{-5}	7	0.5	0.364	0.647	0.276	0.550

Table 6: Validation results on sentiment analysis (SA), paraphrase detection (PD), and semantic textual similarity (STS) for the hyperparameter sweep over learning rate, regularization strength, and dropout. Configurations that encountered invalid arithmetic operations during training have some entries populated with "NaN" (not a number). We observe that the best-performing hyperparameter configuration uses learning rate 1×10^{-4}, regularization strength 5, and dropout rate 0.1.