# Multitask and Task-specific Optimizations for minBERT

Stanford CS224N Default Project

**James Chen,**
Department of Computer Science
Stanford University
jamesc27@stanford.edu

**Krish Parikh**
Department of Mathematics
Stanford University
krishparikh@stanford.edu

## Abstract

We explore various optimizations that can be made to both multitask and task-specific applications of the minBERT model to improve performance on three downstream tasks of sentiment analysis, paraphrase detection, and sentence similarity—taking first place on both the dev and test leaderboards. The primary contrast we investigate is between a single multitask model which is trained on all three tasks and single, task-specific models which are optimized for each task. These task-specific models are expected to do better since they have more targeted training data, but we quantify this differential and explore whether the more generalized knowledge base of the multitask model can compare to concentrated, domain-specific training. We also experiment with additional extensions including an original "center of mass" model layer for semantic textual similarity, an independent implementation of PCGrad gradient surgery for multitask training, an L2 regularization term, some hyperparameter grid searches, some transfer learning steps, and a comparison of different methods for generating sentence-pair embeddings.

## 1 Key Information

- Mentor: Jingwen Wu
- External Collaborators (if you have any): None
- Sharing project: No
- Krish's Contributions: Ownership of specialized training, paraphrase detection task, semantic textual similarity task, and novel center of mass layer.
- James's Contributions: Ownership of multitask training, sentiment analysis task, results analysis, and custom PCGrad gradient surgery implementation.

## 2 Introduction

BERT from Devlin et al. (2018) and GPT-2 from Radford et al. (2019) are both transformer-based models that have shown success on a wide variety of downstream NLP tasks, but each paper takes an opposite approach when training their models for target tasks. Devlin et al. (2018) finetunes a new model for every downstream task attempted, while Radford et al. (2019) finetunes just one model for all downstream tasks attempted. For our project, we compare both strategies—a specialized and multitask model approach—for the downstream NLP tasks of sentiment analysis, paraphrase detection, and semantic textual similarity.
This project also introduces the original "center of mass" model layer for the semantic textual similarity task—motivated by the shortcomings of previous approaches; contains our own implementation of PCGrad gradient surgery for multitask training; and utilizes a combination of training best practices

including hyperparameter grid searches, L2 regularization, and transfer learning to achieve first place on both the dev and test leaderboards.

## 3 Related Work

Our models extend the BERT architecture from Devlin et al. (2018).
**Sentence Pair Inputs:** For both the paraphrase detection and the semantic textual similarity task, our BERT-based model compares two disparate sentence inputs. Devlin et al. (2018) passes these sentence pairs together (delineated by a separator token) to the BERT module, while Reimers and Gurevych (2019) passes each sentence independently to the BERT module and compares their unique representations in a set of final layers. We use the former for specialized models, while we use the latter for generalizability to tasks which involve representations for both single sentences and sentence pairs.
**Training Methodologies:** Ruder et al. (2019) demonstrates that pretraining a task-specific NLP model on abundant data in a related domain can help with generalization to its test set; this strategy is called transfer learning. Kumar et al. (2022) establishes an effective transfer learning procedure, LP+FT, where only the last classification or regression layer of a pretrained model is finetuned on the target task (LP, or linear probing) before the entire model is finetuned on the task (FT, or full finetuning). Yu et al. (2020) experiments with a process of gradient surgery for multitask models in which conflicting gradients are edited. Moreover, L2 regularization methods and their dependency upon other hyperparameters are explored in Lewkowycz and Gur-Ari (2020) as strategies to mitigate overfitting. Yu and Zhu (2020) evaluates different methods of hyperparameter tuning and search space reduction for optimizing convergence when training. We employ a combination of all these strategies in our project.

## 4 Approach

### 4.1 Baseline

Our baseline is a multitask model without the training optimizations mentioned in the Related Work $\rightarrow$ Training Methodologies section. It trains on 30 epochs successively cycling through each of the three downstream tasks to help with generalization to all, and it uses the same loss and architecture as all multitask models we train for this project. We note that our baseline limits the number of batches per epoch on the Quora dataset, since its significantly larger number of examples meant that it would overfit to this task at the expense of others.

### 4.2 Training

#### 4.2.1 Transfer Learning

Because the provided datasets for all three target tasks have a limited number of examples, we use transfer learning—motivated by Ruder et al. (2019)—to achieve better generalizability on the dev and test sets. We pretrain on the Large Movie Review Dataset from Maas et al. (2011) for the sentiment analysis task, and we pretrain on both the Stanford Natural Language Inference Corpus from Bowman et al. (2015) and the Multi-Genre Natural Language Inference Corpus from Williams et al. (2017) for the paraphrase detection and semantic textual similarity tasks. All pretraining is done with the BERT model initialized to the weights for masked language modeling and next sentence prediction from Devlin et al. (2018). We follow Kumar et al. (2022)'s LP+FT procedure for transfer learning. This involves first freezing the BERT model's parameters to finetune just the last layer (LP) before unfreezing all model parameters to finetune the full model (FT).

#### 4.2.2 Gradient Surgery

The baseline, round robin approach was a primitive attempt at preserving gradients for individual tasks during multitask training; PCGrad Gradient Surgery Yu et al. (2020) offered a more sophisticated approach towards this goal. PCGrad works by training the multitask model on all three tasks in each batch and evaluating whether the three associated gradients point in similar directions. We test this using their cosine similarity sign—positive is similar, negative is conflicting. If they do point in

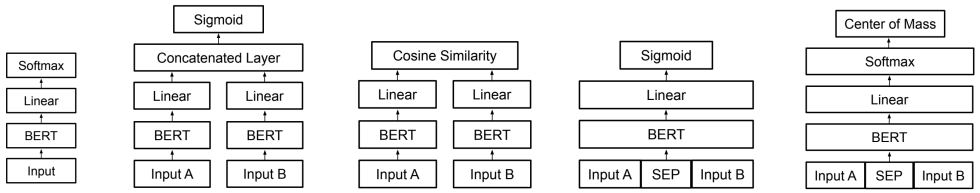
similar directions, we proceed with the updates as normal. But if not, we project each gradient onto the normal plane of different task gradient which conflicts with it. This removes conflicting parts of both gradients to reduce destructive interference.
For any given pair of gradients $g_i$ and $g_j$ which conflict by this cosine similarity metric, we perform the update:

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

This ensures that pairs of conflicting gradients are resolved and eliminates any gradient updates that are highly detrimental for any of our tasks.
Our implementation builds upon code by Tseng (2020), but our application was much more complicated because the final layers for our downstream tasks differ significantly, meaning their gradients weren't directly comparable. To address this, we wrote our own code implementation which separated the shared gradients from the three sets of task-specific gradients and created different optimizers to address each of these groups individually.

### 4.3 Specialized Model Approaches

#### 4.3.1 Architecture




(a) SST (b) Paraphrase with siamese (c) STS with siamese (d) Paraphrase with separator (e) STS with separator

Figure 1: Final Layer Structures

For the paraphrase task and sentence similarity tasks, we experiment with two different methods of applying BERT to create predictions. With the siamese approach, we run both sentences through the BERT model separately and get two different embeddings which are later recombined. In the separator token method, we concatenate the sentences with a separator token and run them together as input to the BERT model. Both methods have extensive precedence in literature, including Reimers and Gurevych (2019) for siamese and Devlin et al. (2018) for separators, which motivated our experimentation. All in all, our multitask training used the siamese approach and our task-specific models used the separator token method.

The `Center of Mass` layer in the semantic textual similarity model is our original contribution. It takes values from six discrete logits normalized to sum to 1 (from the linear and softmax layer) and outputs a continuous value over $[0, 5]$. We describe its motivation and derivation in the next section.

#### 4.3.2 Center of Mass Layer

The `Center of Mass` layer is our unique solution to the discretely rated, yet continuously labeled, SemEval STS Benchmark Dataset from Agirre et al. (2013) for the semantic textual similarity task. Agirre et al. (2013) constructs their dataset from a well-defined six-point scale, ranging from 0 to 5, which allows for interpolations between points to capture uncertainty by annotators. Taking a strict classification approach would neglect these interpolations, but taking a strict continuous approach would ignore the decision boundaries between the six points of the scale. Instead of compromising with either approach, we created the `Center of Mass` layer to do both—use outputs for six distinct logits to generate one continuous value.
We first normalize these six logits to sum to one with softmax. Because these six classes inhabit a linear scale with defined positions and unique weights (their normalized values), we draw inspiration

from the center of mass formula for a linear system of discrete particles to find a continuous output. The equation is as follows:

$$\frac{[l_0, l_1, l_2, l_3, l_4, l_5][0, 1, 2, 3, 4, 5]^T}{sum([l_0, l_1, l_2, l_3, l_4, l_5])},$$

where $l_i$ is the value of a logit.

#### 4.3.3 Loss Functions

**Sentiment Analysis:** We use cross entropy loss for this standard multiclass problem. We considered using a variant of cross entropy loss and mean squared error loss to increasingly penalize sentiment classifications further away from the target class (in-line with the ordinal nature of this classification task), but we settled on cross entropy loss after analysis of early epochs showed that the most classification errors occurred for classes only one point away from the target class.
**Paraphrase Detection:** Because the paraphrase detection task is a binary classification problem which only concerns the probability of a paraphrase (labeled $1$), not the probability of a distinction (labeled $0$), we use binary cross entropy loss for just one logit with a sigmoid-normalized output.
**Semantic Textual Similarity:** We use mean squared error loss, as the outputs are continuous and we seek to increasingly penalize outputs which are further away from the target.

#### 4.3.4 Early Stopping

In all of our models, we implemented early stopping to avoid model overfitting, giving us more adaptable and accurate performance. Generally, we took the model with the best dev performance as our stopping point and ignored further training.

#### 4.3.5 Hyperparameter Search

For the sentiment analysis task, one of the big issues we faced was strong overfitting. We would have training accuracy in the high $0.90$s and dev accuracy hovering a little above $0.50$. We created a hyperparameter search using the grid method detailed in Yu and Zhu (2020) to find the best values for both dropout regularization and L2 regularization to address this issue. We ended up with the best results at $0.4$ dropout and no L2 regularization.

## 5 Experiments

### 5.1 Data

**Task Data:** As described in the default project handout[1], we use the Stanford Sentiment Treebank for sentiment analysis, the Quora dataset for paraphrase detection, and the SEM 2013 STS dataset for semantic textual similarity.
**Transfer Learning Data:** As described in section 4.2.1, we use the Large Movie Review Dataset to pretrain for the sentiment analysis task, and we use both the Stanford Natural Language Inference Corpus and the Multi-Genre Natural Language Inference Corpus to pretrain for the paraphrase detection and semantic textual similarity tasks.

### 5.2 Evaluation method

We use accuracy to evaluate the sentiment analysis and paraphrase detection tasks, and we use the pearson correlation coefficient to evaluate the semantic textual similarity task.

### 5.3 Experimental details

We experimented with various model configurations and hyperparameters, but ended up on a set which seemed to give the best performance for these tasks. Generally, we trained for two epochs with a learning rate of $1e-3$ for the linear probing stage and for fifteen epochs with a learning rate of $2e-5$ for the full finetuning stage. Our batch size was held constant at 16 examples.

[1]Link to default project handout: *https://web.stanford.edu/class/cs224n/project/default-final-project-handout-minbert-spr2024-updated.pdf*

An epoch lasted a minute for sentiment analysis, an hour for paraphrase detection, and a minute for semantic textual similarity. We found that our models reached their best performance per task within 10 epochs for sentiment analysis and semantic textual similarity and within 4 epochs for paraphrase detection.
Generally, we maintained a dropout rate of $0.3$. However, we found significant overfitting on the sentiment analysis task and our hyperparameter search revealed that $0.4$ was more effective for that dataset.

### 5.4 Results

Table 1: dev set performance

| | **SST** | **Paraphrase** | **STS** |
|---|---|---|---|
| Baseline | 0.493 | 0.716 | 0.281 |
| PCGrad | 0.497 | 0.700 | 0.236 |
| Task Specific | **0.548** | **0.903** | **0.897** |

Table 2: best test set performance

| | **SST** | **Paraphrase** | **STS** |
|---|---|---|---|
| Task Specific | **0.544** | **0.903** | **0.9** |

We compare our baseline results to our two different optimized models, the multitask PCGrad model with gradient surgery and our task specific pretrained model with custom loss. In the end, as expected, the task specific datasets had—by far—the best performance. We hypothesize that this is because they could be individually optimized to a very high level and the tasks were different enough with the model being too small to adequately handle each task individually. We also produced confusion matrices that indicate classes where our task specific models performed well and poorly. The PCGrad gradient surgery model actually ended up performing worse than the baseline in two of the three tasks, a phenomenon which we will further discuss in our analysis section.

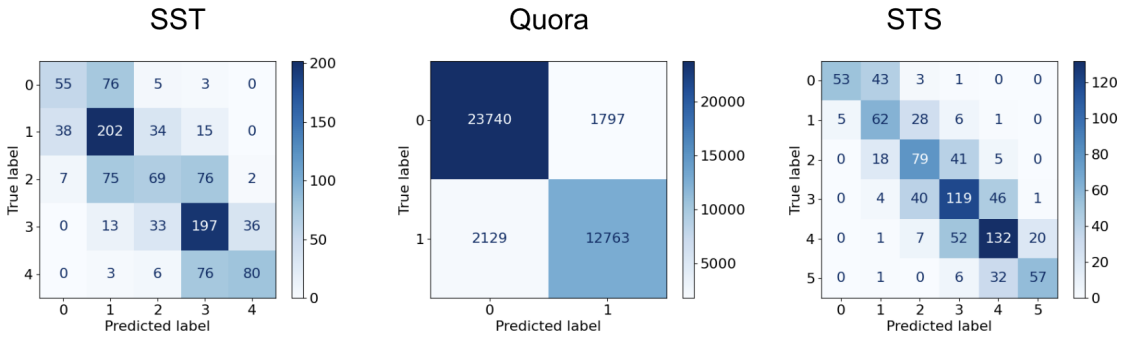



Figure 2: Confusion matrices for task-specific models

There are a few notable observations in the preceding confusion matrices. For the sentiment analysis task, we see that model performance is much better for labels 1 and 3. This indicates that our model is less adept at identifying when things are particularly polarizing or completely neutral, but it could also be related to the fact that the especially positive (0) and especially negative (4) labels had less data. Moreover, we see that the vast majority of incorrect predictions missed the mark by just one point, which is promising since it indicates that when our model is wrong, it is at least close. We find a similar result for the semantic textual similarity task, where nearly all guesses were correct or off by one point—lending confidence to our model's high degree of precision. And for the paraphrase detection task, we see that it is better at predicting sentences which are distinct rather than sentences which are paraphrases. This seems reasonable, since it is intuitive easier to tell whether two concepts are completely different rather than mostly similar. However, there can be cases where sentences have the same overarching meaning but have differences which can be interpreted as significant or not—a much more difficult task.

# 6 Analysis

## 6.1 Gradient Surgery

Surprisingly, our PCGrad method using gradient surgery had worse performance than our baseline. Our hypothesis is that this is because our downstream tasks had much more variety in our multitask applications than the original PCGrad paper. For instance, it may take a very different representation to understand whether a sentence has a positive sentiment rather than if it is similar to another sentence. We evaluate on three quite distinct tasks with different training sets, and we even have a different number of input sentences between tasks. On the other hand, the original PCGrad paper saw improvement in using the same dataset for different tasks (e.g. image classification vs segmentation), and we believe that this different constitutes the poor performance of gradient surgery for this application.

Our explanation for this hypothesis is as follows: in situations with very different inputs, a natural way for the model to orient itself is to allocate certain paths to certain tasks and route different tasks through different parts of the model. However, PCGrad removes these polarizing gradients—preventing the model from addressing the problem in this way. This might provide some insight into why the performance actually decreases when integrating gradient surgery methods.

## 6.2 Performance on SST

We found the low accuracy on SST compared to the other tasks notable. We determined that this may be caused by a few reasons: First, the task is inherently harder. Instead of having binary labels or using correlation metrics like other tasks, we have a five-way classification task measured on accuracy which is punishing for guesses which are just one point away. Another factor is the ambiguity of the ratings by dataset annotators. It can be unclear whether a review is "somewhat positive" or just "positive". While examining where our models went wrong, it was often ambiguous to ourselves as to which was the correct category, and the model classifications seemed quite reasonable—sometimes more than the actual label. This sentiment was frequently supported by other papers working with the same dataset. Therefore, while the accuracy score might seem unimpressive, we believe that it still reflects a relatively strong understanding of sentiment on the part of our model.

## 6.3 Pretraining

Our transfer learning with outside datasets led to significant improvements in model performance. Though there were additional optimizations on top of pretraining our models, pretraining alone gave us strong improvements upon our baseline. For instance, by just pretraining on the Large Movie Review Dataset and introducing zero regularization, our accuracy on the sentiment analysis task increased from $0.493$ to $0.539$. Similarly, our Natural Language Inference pretraining has remarkable benefits on our models' accuracy for the remaining two tasks. This reflects the importance of large amounts of training data for good generalization with BERT. In contrast to our results with PCGrad, it also indicates that while more data is very powerful, if it is too distinct from the target task, it can limit performance gains on smaller models like minBERT.

## 6.4 Center of Mass Layer

Our novel center of mass layer greatly contributed to our strong performance on the semantic textual similarity dataset—rivaling the state of the art results on the same dataset! Alone, integrating this layer instead of the cosine similarity method improved our absolute accuracy by over $0.20$. This validates our hypothesis that it strikes a strong balance in integrating both the discrete and continuous nature of the task labels.

# 7 Conclusion

We conclude that task-specific minBERT models proved more effective than multitask minBERT models on the downstream tasks of sentiment analysis, paraphrase detection, and semantic textual similarity. Upon these distinct tasks with limited training data and small model sizes, concentrating a

model per task enabled better in-task generalizability. We also show that pretraining on large datasets from similar domains to the target task produces models that generalize well, even on limited data tasks. Lastly, we introduce our novel center of mass layer which increases performance on sentence similarity tasks (and perhaps could be used to increase performance on other discrete-yet-continuous labeled tasks), and we build our own implementation of PCGrad gradient surgery.

## 8 Ethics Statement

One important consideration in our work is the differences in how biases is propagated in multi-task versus task-specific datasets. On the one hand, when training a task-specific model, the limited scope of training data may mean that there are specific biases that are common in that dataset which get heavily propagated into the model's prediction process. Meanwhile, since when using multi-task data which is trained on more sources the impacts of these biases might be more dispersed and less prominent in the final model. However, the multi-task approach also has it's own issues with respect to bias. While this dispersion of individual biases seems beneficial, it also makes it much harder to identify and track down the source of. If we take it to the extreme, for a huge model that is trained on a large part of the internet if we discover a bias there is very little we can do to edit the training data since it is so hard to identify where in particular the sentiment is coming from. Instead, we need to make changes and restrictions on the model level. This is not necessarily true of task-specific models where the limited scope can lend itself to more simple data manipulation.

Another point where ethical considerations are particularly important in our research is how these systems could be applied in the real world, especially if released to the public. There is a big difference here, for task-specific models there is a much more contained scope of what users can do with the technology and it is much easier to protect against malicious users. However, once we start making more general multitask models that have strong performance we need to make more careful evaluations about the breadth of use cases. In the end, even though it is the user's responsibility to make responsible use of technologies it is also the job of the creator to make safeguards that protect others from ill use. This is both much more important and much more difficult when using multi-task models than task-specific ones.

To mitigate this first point, the method differs depending on which model structure we are using, multi-task or task-specific. With multitask models, we can build in safeguards using techniques like RLHF to make sure that the model aligns with our understanding of what it should say and doesn't enable people to do dangerous or illegal activities. On the other hand, for task-specific models the task has the potential to be much easier. We can filter through training data, particularly looking at where the problematic or biased behaviors originate from, and eliminate or change them to rework our model from its data.

To address the usage of these systems by the public, we generally just need to be very conscious and thoughtful about what is released to the average user. Extensive testing and evaluation is very important, and safeguards should be put in place to prevent people from using the technology poorly. Again, in relation to what we investigate in this paper it is often easier to release more contained versions of technology that are only trained on specific tasks since it limits the range of potential user activities one needs to consider.

## 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, Atlanta, Georgia, USA. Association for Computational Linguistics.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

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

Ananya Kumar, Aditi Raghunathan, Robbie Jones, Tengyu Ma, and Percy Liang. 2022. Fine-tuning can distort pretrained features and underperform out-of-distribution. *arXiv preprint arXiv:2202.10054*.

Aitor Lewkowycz and Guy Gur-Ari. 2020. On the training dynamics of deep networks with $l\_2$ regularization. *Advances in Neural Information Processing Systems*, 33:4790–4799.

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.

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

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

Sebastian Ruder, Matthew E. Peters, Swabha Swayamdipta, and Thomas Wolf. 2019. Transfer learning in natural language processing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorials*, pages 15–18, Minneapolis, Minnesota. Association for Computational Linguistics.

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

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2017. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*.

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.

Tong Yu and Hong Zhu. 2020. Hyper-parameter optimization: A review of algorithms and applications. *arXiv preprint arXiv:2003.05689*.