Improving Multi-Task BERT Fine-Tuning: Effective Methods and Practices

Stanford CS224N Default Project

Amy Wang
MSE Department
Stanford University
amywy@stanford.edu

Haopeng Xue
CS Department
Stanford University
hxue4@stanford.edu

Xinling Li
EE Department
Stanford University
xinling@stanford.edu

Abstract

This paper investigates the effectiveness of a fine-tuned BERT model across several downstream tasks, including sentiment analysis, paraphrase detection, and semantic textual similarity. We explored preprocessing techniques such as encoding schemes, and fine-tuning approaches like cosine-similarity fine-tuning, PAL, SMART, various loss functions, and other regularized optimization methods. We also examined scheduling methods like annealed sampling, which helps balance the tasks with different data size. Our experiments reveal that the packed encoding scheme significantly improves overall performance. Scheduling methods such as annealed sampling and regularization techniques like SMART and Projected Attention Layer provide moderate enhancements. In addition, our innovative Mixed Cross-Entropy and Log-Based Loss Function benefits sentiment analysis. Our model achieved a score of 0.784 on the dev set and 0.789 on the test set.

1 Key Information to include

- TA mentor: Josh Singh
- External collaborators (if no, indicate "No"): No
- Sharing project (if no, indicate "No"): No
- 9 Late Day: Amy Wang provides 3, Haopeng Xue provides 3, and Xinling Li provides 3

2 Introduction

The introduction of BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al. (2019)) marked a transformative moment in the field of natural language processing (NLP). Using a bidirectional transformer architecture allowed it to capture context from both directions, setting a new standard for performance across various NLP tasks. Since its inception, numerous papers have built upon and extended BERT, exploring its capabilities and enhancing its performance in diverse ways.

In this paper, we aim to investigate the effectiveness of a fine-tuned BERT model across various downstream tasks, including sentiment analysis, paraphrase detection, and semantic textual similarity. Inspired by recent advancements in multitask BERT, we explored preprocessing techniques such as tokenization, encoding schemes, and data augmentation. Our fine-tuning approaches include methods like cosine-similarity fine-tuning, PAL, SMART, various loss functions, and other regularized optimization methods. Additionally, we examined scheduling methods, including annealed sampling. Evaluation metrics such as accuracy and the Pearson correlation coefficient are employed to measure performance, ensuring a comprehensive understanding of BERT's capabilities. Through this comprehensive approach, we aim to establish BERT as a viable solution for diverse NLP tasks, demonstrating its robustness and effectiveness across multiple domains.

3 Related Work

The advent of BERT by Devlin et al. (2019) revolutionized the field of NLP with its novel approach to pre-training deep bidirectional transformers. Unlike its predecessors, BERT reads text bidirectionally, allowing it to capture richer contextual information. This breakthrough has set a new standard for various NLP tasks, leading to numerous extensions and adaptations aimed at enhancing its performance and efficiency. The application of BERT in multitask learning has garnered significant attention due to its ability to leverage shared representations across various NLP tasks. BERT's architecture, which captures contextual information bidirectionally, is particularly well-suited for multitask applications where understanding nuanced language dependencies is crucial. Several studies have explored and extended BERT's capabilities in this domain, demonstrating its versatility and effectiveness. MT-DNN (Liu et al. (2019)) first integrates BERT into a multitask deep neural network. By jointly training BERT on multiple NLP tasks, including question answering, sentiment analysis, and natural language inference, MT-DNN achieved state-of-the-art performance across several benchmarks. The shared representations learned by BERT improved the model's generalization across tasks, highlighting the benefits of multitask learning. Furthermore, several works have attempted to enhance BERT's multi-task capabilities. Projected Attention Layers (PALs), as introduced in the Stickland and Murray (2019) work by projecting the attention mechanism into a lower-dimensional space, thereby reducing computational overhead and improving model efficiency. This technique allows for more effective parameter sharing across tasks, facilitating better generalization and faster adaptation to new tasks. ELECTRA (Clark et al. (2020)) introduced a novel pre-training method that, when combined with multi-task learning, resulted in more efficient and effective training. By replacing masked tokens with plausible alternatives, ELECTRA's generator-discriminator approach enhanced learning efficiency, showcasing the potential of innovative pre-training strategies in multi-task frameworks.

4 Approach

4.1 Baseline Model

The baseline model is the original BERT model Devlin et al. (2019). BERT converts the input sentence into embeddings, comprising token, segmentation, and position embeddings, which are processed through 12 Encoder Transformer layers. Each input is represented by the pooled representation from the hidden state of the [CLS], a special token at the beginning of each input sequence. For the three downstream tasks, we add a linear layer for sentiment analysis and semantic textual similarity, and two linear layers for paraphrase detection. The model is optimized using the Adam Optimizer with Decoupled Weight Decay Regularization Kingma and Ba (2017); Loshchilov and Hutter (2019).

4.2 Pairs Encoding Scheme: Separate-Encoding, and Packed-Encoding

Both paraphrase detection and similarity tasks require encoding pairs of sentences to determine their relationship. We explored two encoding methods 1: (1) Separate-Encoding: For semantic textual similarity, we use the method in Reimers and Gurevych (2019), which computes similarity scores based on the embeddings of the two sentences. For paraphrase detection, we concatenate the embeddings of the two sentences before feeding them into the linear layer. (2) Packed-Encoding: Following the original BERT model Devlin et al. (2019), we pack the two sentences with a [SEP] token between them and represent the pair using the hidden vector of the [CLS] token.

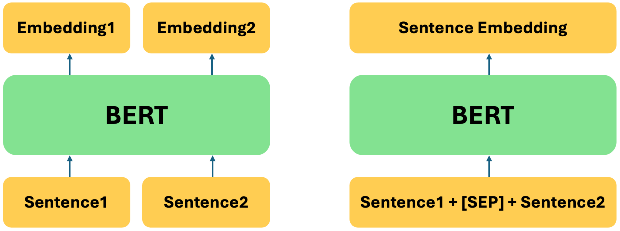

Figure 1: Pairs Encoding Scheme: Separate-Encoding (Left) and Packed-Encoding Right)

4.3 Cosine-Similarity Fine-Tuning

We use cosine-similarity fine-tunning for STS and Paraphrase Detection task for separate encoding. The cosine similarity score measures the cosine of the angle between our two embeddings, which could tell us whether these two embedding are "close" or "far" form each other, and we will get a similarity scores within [-1, 1] range.

$$\text{score} = \frac{\mathbf{p} \cdot \mathbf{q}}{\|\mathbf{p}\|_2 \|\mathbf{q}\|_2}$$

STS: We apply mean-pooling strategy to compute the average of the token embeddings, weighted by the attention mask. Next, we calculate the cosine similarity between the two embeddings. To evaluate the model's performance, we compute the Mean Squared Error (MSE) loss between the predicted cosine similarity scores and the actual labels. This loss function measures how closely the predicted similarity scores align with the target values.

4.4 Mixed Cross-Entropy and Log-Based Loss Function for Sentiment Analysis

While cross-entropy is typically used for classification tasks, it doesn't account for the ordinal relationship between classes. For sentiment analysis, we have 5 classes, from 0 to 4, ranging from negative to positive. Due to the ordinal nature of the classes, cross-entropy is not optimal because it doesn't incorporate this ordinal characteristic into its feedback. Castagnos et al. (2022) proposes an ordinal log-loss (OLL):

$$L_{OLL-\alpha}(P, y) = -\sum_{i=1}^{N} \log(1 - p_i) d(y, i)^{\alpha}$$

where N is the number of classes, p_i is the output probability for the ith class, and $d(y, i)$ is the distance between the true label and the ith label. In our case, $d(y, i) = |y - i|$. α is a positive hyper-parameter. The greater α is, the higher the loss function is when the distance between the output predictions and the labels is large. It is set to 1.5 according to the experimental results in Castagnos et al. (2022).

Although this method considers the ordinal relationship between labels, compared to cross-entropy, it is less effective at pushing the label to the correct class. Therefore, we use an innovative mixed loss function:

$$L_{CEOLL} = L_{CE} + L_{OLL-1.5}$$

This approach considers the ordinal relationship between labels while ensuring the prediction is as close as possible to the true label.

4.5 Multiple Negatives Ranking Loss Learning

We implemented the Multiple Negatives Ranking Loss function, described in Efficient Natural Language Response Suggestion for Smart Reply Henderson et al. (2017). To implement this loss function, we use all positive examples in the Quora dataset, so that we can make sure that we only have (anchor, positive) pairs. Then we compute the cosine similarity between each anchor a_i and every positive embedding p_j in the batch to get a N x N similarity matrix. In this similarity matrix, the diagonal entries represent the cosine similarity between correct (anchor, positive) pairs, such as cos(ai, pi). The off-diagonal entries represent the cosine similarity between ai and pj, and pj is treated as the negatives for anchor ai. Then Our model can be more robust because it is trained to distinguish the positive from a variety of negatives.

$$score = \begin{bmatrix} \cos(a_1, p_1) & \cos(a_1, p_2) & \cdots & \cos(a_1, p_n) \\ \cos(a_2, p_1) & \cos(a_2, p_2) & \cdots & \cos(a_2, p_n) \\ \vdots & \vdots & \ddots & \vdots \\ \cos(a_n, p_1) & \cos(a_n, p_2) & \cdots & \cos(a_n, p_n) \end{bmatrix}$$

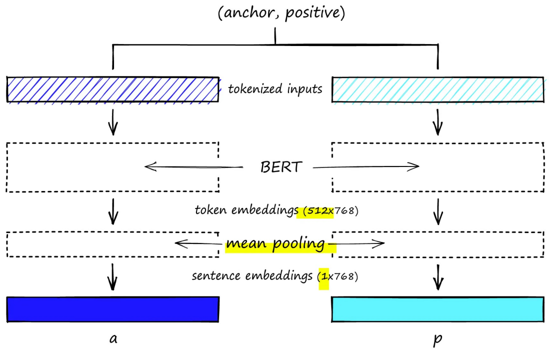

Figure 2: Multiple Negatives Ranking Loss Learning Briggs

4.6 Contrastive Learning

We implemented the contrastive learning on Paraphrase Detection. According to Gao et al. (2021), they introduced simple unsupervised contrastive learning, which is "Simple Contrastive Learning of Sentence Embeddings" (SimCSE). SimCSE seeks to enhance performance by drawing semantically similar items closer and distancing those that are dissimilar. We filtered out the negative examples in our dataset and only keep (anchor, positive), which represent by (x_i, x_i^+).

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

where h_i and h_i^+ denote the representations of x_i and x_i^+ , τ is a temperature hyperparameter, and sim(h_i, h_i^+) is the cosine similarity.

4.7 Gradient Surgery

One challenge for multi-task learning is conflicting gradients, which means optimizing one task might weaken the performance of another task. Yu et al. (2020) proposes gradient surgery, a method that projects a task's gradient onto the normal plane of the gradient of any other task that has a conflicting gradient (see Figure 3). The modified gradient is calculated as $g_i = g_i - \frac{g_i \cdot g_j}{||g_j||^2} \cdot g_j$.

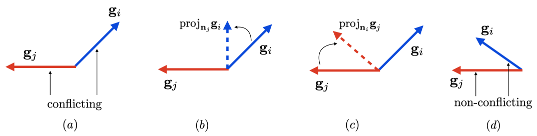

Figure 3: Conflicting Gradients

4.8 SMART: Smoothness-Inducing Regularization

To mitigate overfitting when fine-tuning complex models with limited data, Jiang et al. (2020) proposes Smoothness-Inducing Adversarial Regularization (SMART), which adds a regularizer to the standard loss function:

$$\min_{\theta} F(\theta) = L(\theta) + \lambda_s R_s(\theta)$$

$$R_s(\theta) = \frac{1}{n} \sum_{i=1}^{n} \max_{||\xi' - \xi|| \leq \epsilon} l_s(f(x_i; \theta), f(x_i; \theta))$$

The regularizer ensures that the output does not change significantly when there are perturbations in the input. Our code is implemented based on archinetai (2022). We only tune λ_s to adjust the weight of the regularizer and use the default values for other hyperparameters. For sentiment analysis and paraphrase detection, l_s is chosen as the symmetrized KL-divergence. For semantic textual similarity, we use the mean square error loss.

4.9 Annealed Sampling

Annealed Sampling, proposed in Stickland and Murray (2019), is a method used to schedule the training of multi-task learning models in a way that balances the training across tasks with varying amounts of data. The goal is to avoid over fitting tasks with smaller datasets and under-training tasks with larger datasets by adjusting the sampling probability of each task over time. Tasks are initially sampled in proportion to their training set size, meaning tasks with more data are sampled more frequently. This helps ensure that each task is adequately represented during training. As training progresses, the influence of the training set size on the sampling probability is gradually reduced. This is achieved by using an annealing factor that decreases over epochs, allowing the model to focus more equally on all tasks towards the end of training. The annealing factor is defined as:

$$\alpha = 1 - \frac{0.8}{e-1} * \frac{E-1}{E-1}$$

where α is the annealing factor, e is the current epoch, and E is the total number of epochs. This method helps mitigate the risk of interference between tasks and improves overall model performance.

4.10 Projected Attention Layer (PAL)

PAL (Stickland and Murray (2019)) IS introduced as an efficient adaptation module for multi-task learning with the BERT model. The key idea behind PALs is to add a low-dimensional multi-head attention layer in parallel with the existing BERT layers. This allows the model to share a significant portion of its parameters across multiple tasks while maintaining a small number of task-specific parameters for adaptation. The PALs are designed to address the challenge of efficiently adapting a single pre-trained model to perform well on various tasks, especially when fine-tuning separate models for each task would be impractical due to computational or storage constraints. PALs consist of a low-dimensional multi-head attention layer added in parallel to the normal BERT layers. This attention layer operates on a lower-dimensional space, allowing for efficient computation and parameter sharing. By using a low-rank approximation for the key operations of the model, PALs significantly reduce the number of additional parameters required for each task. Specifically, PALs can achieve comparable performance to fully fine-tuned models with approximately seven times fewer parameters. PALs are integrated into the BERT model by adding a task-specific function in parallel with the self-attention layers. This allows the model to adapt its representations for each specific task without significantly increasing the parameter count.

5 Experiments

5.1 Data

Stanford Sentiment Treebank Socher et al. (2013): The SST dataset consists of sentences extracted from movie reviews, with each sentence labeled with a sentiment ranging from 0 (negative) to 4 (positive). **Quora** Csernai: Quora's dataset consists of pairs of questions, with a label of 0 or 1 indicating if the questions are paraphrases of each other. **SemEval STS Benchmark Dataset** Socher et al. (2013): The STS dataset consists of pairs of sentences with a similarity score between 0 (not at all related) to 5 (same meaning). The detailed data statistics are shown in 1.

Dataset	Task	Train Size	Dev Size	Test Size
SST	Sentiment Analysis	8544	1101	2210
Quora	Paraphrase Detection	283010	40429	80859
STS	Semantic Textual Similarity	6040	863	1725

Table 1: Dataset Statistics

5.2 Evaluation method

The evaluation method for sentiment analysis and paraphrase detection is accuracy, calculated as the percentage of correctly classified instances. For Semantic Textual Similarity, we use the Pearson correlation coefficient to measure the linear correlation between model predictions and ground truth.

5.3 Experimental details

Unless stated otherwise, the learning rate is fixed at 1e-5. The batch size for Sentiment Analysis and Semantic Textual Similarity is 8, and 16 for Paraphrase Detection. The dropout probability is set to 1e-3, and the number of epochs is 10. The AdamW optimizer is used with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\lambda = 1e - 3$. During training, batches for the three tasks are alternated. By default, the batch size is set to the minimum of the three tasks' batch sizes. However, the batch size is set to the maximum when the epoch number is divisible by 4.

5.4 Pairs Encoding Scheme

Since the performance of Packed-Encoding is significantly higher than Separate-Encoding, we use Packed-Encoding by default for the rest of the model. 2

Model	Para Dev Acc	STS Dev Acc
Separate-Encoding	0.657	0.266
Packed-Encoding	**0.900**	**0.809**

Table 2: Performance Comparison of Two Pair Encoding Schemes

5.5 Loss Function for Sentiment Analysis

As expected, using OLL alone doesn't have improvement, but there is a 0.03 improvement when using the mixed loss function.

Model	SST Dev Acc
Cross-Entropy	0.526
OLL	0.516
CEOLL	**0.529**

Table 3: Comparison of SST Dev Accuracy with Different Loss Functions

5.6 Results on Leaderboard

The experiment results reveal several key insights into the performance of various models and techniques across different NLP tasks. **(1)** Contrastive loss did not perform as expected. Although it achieved nearly 0.98 accuracy on the training dataset for paraphrase detection, it only reached 0.64 on the dev set. Doubling the dropout probability reduced overfitting on the Quora dataset but decreased accuracy on the STS and SST datasets. Adjusting the temperature parameter τ revealed that higher τ values initially improved accuracy but declined over epochs. We set τ to 0.1. **(2)** The use of cosine similarity on the STS dataset underperformed. Reducing the Quora training data improved STS performance to about 70%, suggesting dataset imbalance was an issue. The MNRL loss function also resulted in low accuracy on the STS dataset. This is because the MNRL loss function is effective for training data with (anchor, positive) pairs labeled as 0 or 1. However, our STS labels are floating-point numbers, which likely caused this issue. **(3)** Separate encoding (SE) consistently underperformed compared to packed encoding. Extensions like MNRL, Cosine Similarity, and SimCSE did not significantly enhance overall performance compared to the base model. **(4)** Gradient Surgery did not improve overall performance as expected. Despite identifying conflicting gradients between the sentiment task and other tasks, it did not enhance sentiment accuracy and only slightly improved similarity performance. This lack of improvement might be due to the complexity of effectively projecting gradients in a way that benefits all tasks simultaneously. **(5)** Both PAL and annealed sampling techniques showed expected improvements over the base model across all tasks, indicating their benefit in balancing training dynamics. **(6)** The combination of SMART and CEOLL was the best-performing approach, significantly increasing sentiment accuracy and improving performance in other tasks. This demonstrates the effectiveness of combining regularization techniques with advanced loss functions to enhance model robustness and performance.

Model	SST Dev Acc	Para Dev Acc	STS Dev Acc	Overall Score
Base Model (SE)	0.389	0.657	0.266	0.559
Base Model+MNRL (SE)	0.262	0.672	0.039	0.486
Base Model+Cosine Similarity (SE)	0.128	0.705	0.467	0.517
Base Model+SimCSE (SE)	0.262	0.637	0.045	0.474
Base Model	0.505	**0.900**	0.809	0.770
Best Model+Gradient Surgery	0.480	0.897	0.820	0.762
Best Model+PAL+Annealed	0.513	0.889	0.830	0.772
Best Model+SMART+CEOLL	**0.519**	0.899	**0.868**	**0.784**

Table 4: Comparison of models on SST, Paraphrase, and STS dev sets with overall scores

Model	SST Test Acc	Para Test Acc	STS Test Acc	Overall Score
Best Model	0.533	0.898	0.869	0.789

Table 5: Model Performance on SST, Paraphrase, and STS test sets with overall score

6 Analysis

The incorrect examples for the three tasks can be found in the Appendix.

For sentiment analysis, the confusion matrix (Figure 4) shows a concentration along the diagonal, gradually decreasing towards the off-diagonal elements, indicating that the majority of predictions are correct. The most common errors occur between adjacent labels, while misclassifications between negative and positive sentiments are rare. Upon examining the incorrect examples, the model fails to identify finer-grained sentiments and complex sentence structures. For example, "A sober and affecting chronicle of the leveling effect of loss," is misclassified as sentiment 3 when it is actually sentiment 2. This suggests that the model may struggle with implicit or complex emotional expressions. Similarly, "Holden Caulfield did it better" is predicted as sentiment 3 instead of sentiment 1, indicating difficulty in understanding literary or metaphorical language.

For paraphrase detection, the analysis of the incorrect examples has shown that the model fails to capture subtle nuances in language and understand variations in sentence structure. For example, the sentences "What would Hillary Clinton do as the president?" and "How would Hillary Clinton be as a president?" are incorrectly classified as paraphrases, possibly due to the large overlap between sentences instead of identifying the subtle differences in meaning between "what" and "how". Another example, "Which is the most reliable car brand in India?" and "Which is the most car reliable company in India?" are also incorrectly classified as non-paraphrases. This suggests that the model struggles with identifying similar word meanings. In another example, the sentences "What could cause a period to be 7 days late?" and "What can cause a woman's period to be late?" are incorrectly classified as paraphrases. This could be due to the model's inability to differentiate the subtle differences and additional information in sentences.

For semantic textual similarity task, Figure 5 indicates a predominantly linear relationship between ground truth scores and predictions, with very few outliers. This suggests that the model generally performs well in assessing the degree of semantic equivalence between texts. However, an examination of the incorrect examples reveals that the model may focus too much on the words and structure instead of the meaning. For instance, in the sentence pair "Work into it slowly." and "It seems to work.", the model assigns a similarity score of 2.2, indicating some level of relatedness, whereas the ground truth score is 0.0, suggesting no similarity. This discrepancy suggests that the model may not adequately capture the subtle nuances in meaning between these sentences. Similarly, in the sentence pair "Democracy is a threat to liberty." and "Democracy has nothing to do with liberty.", the model assigns a similarity score of 3.5, whereas the ground truth score is 1.8. This indicates that the model may struggle with sentences that express different ideas but in similar ways. Therefore, we may need to adjust the model's attention to focus more on the meaning instead of words and structure.

Overall, the lower accuracy in the sentiment analysis task is primarily due to the 5 classes for classification. While the model can effectively distinguish between positive and negative sentiments, further improvements are needed to differentiate between more nuanced levels of sentiment intensity.

The performance in paraphrase detection and similarity tasks is generally good. However, errors in these tasks are mainly attributed to the model's excessive focus on the specific words used and the structure of the sentences, rather than capturing the correlation between words and understanding the overall meaning of the sentences.

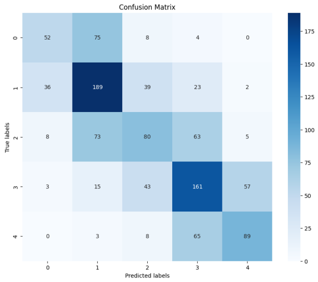

Figure 4: Confusion Matrix for Sentiment Analysis

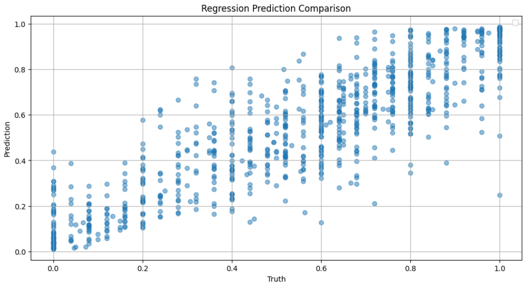

Figure 5: Similarity Task Visualization

7 Conclusion

In conclusion, our study explored 9 techniques to enhance multi-task learning across three tasks. The packed encoding scheme significantly improved overall performance, and methods like annealed sampling, SMART, and Projected Attention Layer provided notable enhancements. Our innovative Mixed Cross-Entropy and Log-Based Loss Function proved beneficial for sentiment analysis. However, techniques like Gradient Surgery and separate encoding methods like Cosine Similarity and MNRL did not perform as expected. Our best model utilized SMART regularization with the CEOLL loss function, achieving competitive results with a score of 0.784 on the development set and 0.789 on the test set. Despite these achievements, a key limitation of our study was the lack of sufficient time, GPU resources, and memory to explore other learning extensions. Moving forward, there is room for improvement, particularly in fine-grained sentiment analysis and in enhancing the model's understanding of contextual meaning over individual words and sentence structures. Future research could focus on exploring more advanced methods for sentiment analysis and integrating external knowledge sources to further enhance model performance.

8 Ethics Statement

1. Our model is trained on the corpus that are typically in English, so it might not only perform poorly on other languages but could also contribute to cultural homogenization. This can lead to a loss of linguistic diversity as minority languages may be underrepresented or misrepresented. **Mitigation Strategy:** We could add multilingual dataset and culturally diverse examples in our dataset. Also, we could explore architectures designed for handling multiple languages. This may help the model understand the nuances of different languages better.

2. Especially for the sentiment analysis task, our model just simplifies the complex emotions into three categories (positive, negative and neutral). This can miss the subtleties of a human's response to a movie, potentially leading to misunderstandings. **Mitigation Strategy:** Instead of just classify the movie review into three categories, we could try to expand our category set that might include: "angry", "sad" etc. Also, a human review process may be helpful. The outputs of the model can be periodically reviewed and corrected by human annotators. This can help the model learn from complex cases and reduce errors due to oversimplification.

References

archinetai. 2022. smart-pytorch. `https://github.com/archinetai/smart-pytorch/tree/main`.

James Briggs. Next-gen sentence embeddings with multiple negatives ranking loss | pinecone.

François Castagnos, Martin Mihelich, and Charles Dognin. 2022. A simple log-based loss function for ordinal text classification. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4604–4609, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators.

Kornél Csernai. First quora dataset release: Question pairs. `https://quoradata.quora.com/First-Quora-Dataset-Release-Question-Pairs`.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. *CoRR*, abs/2104.08821.

Matthew Henderson, Rami Al-Rfou, Brian Strope, Yun-Hsuan Sung, László Lukács, Ruiqi Guo, Sanjiv Kumar, Balint Miklos, and Ray Kurzweil. 2017. Efficient natural language response suggestion for smart reply. *ArXiv*, abs/1705.00652.

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

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019. Multi-task deep neural networks for natural language understanding.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization.

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

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew 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, Seattle, Washington, USA. Association for Computational Linguistics.

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

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

A Appendix

Task	**Sentence(s)**	**Truth**	**Prediction**
Sentiment	Seldom has a movie so closely matched the spirit of a man and his work.	4	3
Sentiment	A sober and affecting chronicle of the leveling effect of loss.	2	3
Sentiment	It 's like watching a nightmare made flesh	0	1
Sentiment	Holden Caulfield did it better.	1	3
Paraphrase	What would Hillary Clinton do as the president? How would Hillary Clinton be as a president?	0	1
Paraphrase	Which is the most reliable car brand in India? Which is the most car reliable company in India?	1	0
Paraphrase	What could cause a period to be 7 days late? What can cause a woman's period to be late?	0	1
Similarity	Work into it slowly. It seems to work.	0.0	2.2
Similarity	A man is standing on top of a rock or mountain watching the sun set. A man standing in the mountains watching a sunset.	4.8	3.2
Similarity	Democracy is a threat to liberty. Democracy has nothing to do with liberty.	1.8	3.5
Similarity	Syria opposition threatens to quit talks. Syria opposition agrees to talks.	2.0	4.1

Table 6: Incorrect Examples of Three Tasks