From BERT to Brilliance: An Analytical Approach to Advancing Multitask Learning for NLP

Stanford CS224N Default Project

Akea Pavel
Department of Computer Science
Stanford University
apavel17@stanford.edu

Adrian Mendoza-Perez
Department of Computer Science
Stanford University
mendy@stanford.edu

Abstract

Pre-trained language models like BERT have revolutionized natural language processing tasks, yet their ability to grasp nuanced language features remains an ongoing challenge. In this paper, we present a series of techniques to significantly improve the performance of a pre-trained language model, minBERT, on three downstream NLP tasks: sentiment analysis, paraphrase detection, and semantic textual similarity. Specifically, we modify Sentence-BERT (SBERT), a version of BERT that employs siamese and triplet network structures to derive semantically meaningful sentence embeddings that can be compared using cosine similarity. Our approach incorporates a mixture of cross-encoding and bi-encoding BERT's input sentences, employs MNRL to enhance the model's ability to distinguish between similar and dissimilar pairs, and applies Ordinal Log-Loss to mitigate errors with adjacent boundaries. These combined strategies yield substantial improvements in multitask learning ability, achieving a overall test score of 0.650 on the test set and surpassing the SBERT baseline.

1 Key Information to include

Our mentor is Johnny Chang (cjohnny@stanford.edu) and we are not sharing this project with any other classes nor do we have any external collaborators. Akea and Adrian worked together to implement the initial $BERT_{\text{Base}}$ model. Akea then implemented the S-BERT-inspired model and Adrian followed up by implementing Multiple Negatives Ranking Loss and Ordinal Log-Loss. Akea and Adrian both contributed equally towards writing the paper.

2 Introduction

In recent years, significant advancements in natural language processing (NLP) have been driven by pre-trained models such as BERT (Devlin et al., 2019). However, applying BERT to practical tasks often requires fine-tuning for specific downstream applications, which presents challenges like computational inefficiency and poor embeddings. BERT's method of processing sentence pairs by feeding both sentences into the network simultaneously results in a dramatic increase in computation. For tasks like semantic textual similarity (STS), this approach becomes computationally prohibitive, requiring around 50 million inference computations for 10,000 sentences, translating to roughly 65 hours on a modern GPU (Reimers et al., 2019). Moreover, BERT's approach of averaging the output layer (known as BERT embeddings) or using the output of the first token (the [CLS] token) often results in poor sentence embeddings, which are frequently worse than those obtained by averaging GloVe embeddings (Pennington et al., 2014).

To address these challenges, we modify Sentence-BERT (SBERT), a BERT variant designed to derive semantically meaningful sentence embeddings using siamese and triplet network structures, to improve $BERT_{\text{Base}}$ performance in sentiment classification, paraphrase detection, and semantic textual

similarity. SBERT efficiently compares embeddings using cosine similarity, reducing $BERT_{\text{Base}}$ computational complexity, improving its sentence embeddings, and cutting down its processing time for tasks like semantic similarity search from hours to seconds. We also propose a multitask learning strategy that incorporates several key techniques to improve SBERT, such as Bi-encoding and Cross-encoding to improve sentence embeddings; Multiple Negatives Ranking Loss (MNRL) to enhance the model's ability to distinguish between similar and dissimilar pairs, crucial for tasks like paraphrase detection and STS; Ordinal Log-Loss to enhance the boundaries between adjacent labels. Our goal is to create a model that utilizes the power of pre-trained BERT while addressing the unique challenges of multitask learning. Through our experiments and analysis, we showcase the effectiveness of our approach, highlighting the strengths and weaknesses of our model.

3 Related Work

Our work begins with an implementation of the $BERT_{\text{Base}}$ introduced by Devlin et al. (2019), which implements a transformer-based model capable of achieving state-of-the-art results on a range of natural language processing tasks through a novel approach of bidirectional training. Despite its significant contributions, BERT exhibits notable limitations, particularly in handling domain-specific language and requiring substantial computational resources for training and fine-tuning. These limitations provide the starting point for our enhancements aimed at optimizing the efficiency and applicability of the model on three downstream tasks, sentiment analysis, paraphrase detection, and semantic textual similarity.

To address some of BERT's limitations, particularly its performance in semantic similarity assessment, we drew from ideas implemented within Sentence-BERT (SBERT) by Reimers and Gurevych (2019). SBERT modifies the original BERT architecture to produce fixed sentence embeddings that can be compared using cosine similarity. This adaptation reduces the computational cost for tasks involving semantic comparisons of sentences, thus alleviating the need for pairwise comparisons inherent in BERT's original framework.

We continued to fine-tune our model through an evaluation of encoding strategies, testing the effectiveness of cross-encoding as implemented in the original BERT architecture, against the cosine similarity approach used by SBERT. Cross-encoding allows for the processing of sentence pairs together in a single pass. This can allow the model to detect inter-sentence nuances directly. Cosine similarity assessments involve the separate processing of each sentence, and focusing instead on efficiency, potentially missing out on nuance within sentence pairs (Reimers and Gurevych, 2019).

Furthermore, to refine our approach for the tasks involving sentence pairs, paraphrase detection and semantic textual similarity, we integrated the Multiple Negatives Learning Rank (MNLR) method described by Henderson et al. (2017). This technique employs multiple negative samples, enhancing the model's discriminatory capabilities. Within a single batch, there is a single positive pair and multiple negative pairs. MNLR aims to minimize the approximated mean negative log probability of the data, optimizing the similarity measure between the positive pair and maximizing the differences with the negative pairs. We utilized cosine similarity as our scoring function in this process, using its effectiveness in measuring semantic similarity.

Additionally, to improve the adjacent boundaries between labels in the sentiment classification task, we implemented Ordinal Log-Loss (OLL). Traditional cross-entropy loss does not fully capture the ordinal nature of the labels. Therefore, we explored OLL as introduced by Castagnos et al. (2022), which not only encourages correct predictions but also penalizes predictions that are far from the true label.

4 Approach

4.1 Baseline

For our default model, we implemented Bidirectional Encoder Representations from Transformers (BERT) with pre-trained weights.

4.1.1 BERT Implementation

We implemented the BERT model, focusing on the multi-head self-attention and transformer layers as described in the original BERT paper. The bidirectional encoder representations allow the model to capture valuable contextual information from both directions. BERT consists of 12 transformer layers, each with 768 hidden units and 12 attention heads, designed to maintain a balance between efficiency and performance. The attention mechanism is defined as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

We initialized BERT with pre-trained weights and fine-tuned it on the SemEval dataset to adapt it for our specific tasks.

4.1.2 Adam Optimizer

To train our model efficiently, we implemented the Adam optimizer, which computes adaptive learning rates for different parameters by estimating the first and second moments of the gradients. The optimizer adjusts the learning rate for each parameter based on these estimates. It also includes bias correction to ensure stability in the early stages of training, and a weight decay regularization technique to prevent overfitting. The update rule for the Adam optimizer is given by:

$$\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \frac{m_t/(1-\beta_1^t)}{\sqrt{v_t/(1-\beta_2^t)} + \epsilon}$$

4.2 S-Bert With Additional Layers

Inspired by the architecture implemented in S-BERT, we wanted to evaluate how different pooling methods and cosine similarity fine-tuning could make an improvement in our downstream tasks as was demonstrated by Reimers and Gurevych (2019).

4.2.1 Sentiment Analysis Architecture

We used a simple architecture for sentiment analysis, inputting a tokenized sentence into the BERT model, producing a 768-dimensional embedding as shown in Figure 3 in the Appendix. The key [CLS] token embedding, representing the entire sentence, is chosen with a CLS pooling strategy. This embedding then goes through dropout for regularization, passes through a linear projection layer to output unnormalized sentiment probabilities across five classes, and is normalized using the sigmoid function. Then, the cross-entropy loss is computed with true labels. Finally, a prediction is made by taking the argmax of the normalized probabilities.

4.2.2 Semantic Textual Similarity Architecture

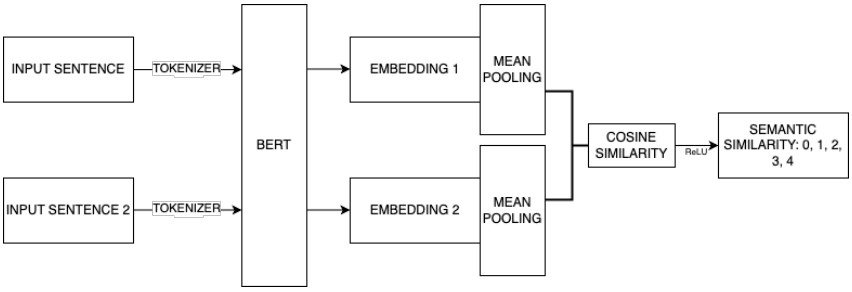

Figure 1: Semantic Textual Similarity Architecture

For the STS architecture, we were inspired by the Siamese network structure implemented in S-BERT by Reimers and Gurevych (2019). In this process, each input is fed to the same model in order to produce two embeddings, which are then averaged with mean pooling. Then, we calculate the cosine

similarity between the averaged embeddings, normalize the output with a ReLU function to change the range to [0, 1], then multiplied by 5 to scale to [0, 5] to ensure predictions fall in the desired range.

4.2.3 Paraphrase Detection Architecture

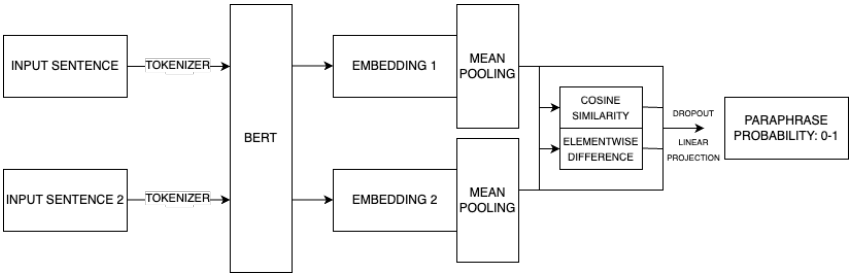

Figure 2: Paraphrase Detection Architecture

As paraphrase detection combines parts of both sentiment analysis and STS, we were inspired by S-BERT to create a architecture using use a linear projection layer with dropout to determine the binary classification of whether two input sentences are paraphrases. The input to this layer includes the embeddings of both sentences, their cosine similarity, and their absolute element-wise difference. The weight matrix of the projection layer is dimensionally $W \in \mathbb{R}^{3\times768+1\times1}$. To generate a binary prediction, the output from the linear layer is normalized with a sigmoid function and rounded to the nearest integer.

4.3 Cross-Encoding, Cosine Similarity, and Bi-Encoding

As STS and Paraphrase Detection are sentence-pair tasks, this can present a challenge for the BERT model, which is traditionally limited to processing single input sequences. To try to improve performance on these sentence-pair tasks, we explored the concepts of cross-encoding and bi-encoding with cosine similarity, as mentioned in the architecture in section 4.2. The alternative approach of cross-encoding, used by Devlin et al. (2019) in the original BERT model, merges each input sequence with a [SEP] token before processing. Then, a classification/regression layer is used on the output embedding for a prediction as shown in Figure 4a in the Appendix. As opposed to Figure 4b in the Appendix, which uses the mean-pooled BERT embeddings for two input sequences and calculates their cosine similarity. The similarity score is then normalized using a ReLU activation and scaled by a factor of 5 before being returned.

4.4 Multiple Negative Loss Ranking (MNRL) for Paraphrase

To improve performance particularly on paraphrase detection, due to the sentence-pair structure, we implemented the Multiple Negatives Ranking Loss (MNRL) as referenced in Henderson et al. (2017). This loss function aims to minimize the distance between embeddings of similar sentences while maximizing the distance between embeddings of dissimilar sentences. The objective function is defined as:

$$J(x, y, \theta) = -\frac{1}{K}\sum_{i=1}^{K} \log P_{\text{approx}}(y_i|x_i) = -\frac{1}{K}\sum_{i=1}^{K}\left[S(x_i, y_i) - \log\left(\sum_{j=1}^{K} e^{S(x_i,y_j)}\right)\right]$$

4.5 Ordinal Log-Loss for Fine-Grained Sentiment Analysis

Given that the initial loss function for sentiment analysis was the cross-entropy loss function, there was no accounting for the ordered nature of the sentiment classes, treating the task solely as categorical classification. In this case, Ordinal Log-Loss was helpful in penalizing prediction not only based off classification accuracy, but also by how far off they were in terms of order. We implemented the Ordinal Log-Loss as described by Castagnos et al. (2022) and took inspiration from

glanceable-io, which is shown here:

Given a set of classes $C_1, C_2, \ldots, C_N$, and the ordinal predictions $P = (p_1, \ldots, p_N)$ for these classes, the loss is defined as:

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

where $d(y, i)$ represents the distance between the true label y and the class C_i, and α is a hyper-parameter that adjusts the sensitivity of the loss to the distance between the predicted and actual labels.

5 Experiments

5.1 Data

We used four datasets for our experiments: the Stanford Sentiment Treebank (SST) for sentiment classification, the Quora Question Pairs dataset for paraphrase classification, the SemEval Semantic Textual Similarity (STS) Benchmark dataset for similarity classification, and the positive sentence pairs of the Quora Question Pairs for MNRL. The SST dataset contains sentences from movie reviews with sentiment labels, split into 8,544 training, 1,101 validation, and 2,210 test examples. The Quora dataset contains 404,298 question pairs with labels indicating whether they are paraphrases, split into 283,010 training, 40,429 validation, and 80,859 test examples. The SemEval STS dataset contains 8,628 sentence pairs with similarity scores, split into 6,040 training, 863 validation, and 1,725 test examples. The positive sentence pairs dataset contains 13073 training examples.

5.2 Evaluation method

To evaluate our model, we used the metrics as defined in the project handout. For the sentiment analysis and paraphrase detection tasks, we used a simple accuracy score to compare the true and predicted labels. For the semantic textual similarity task, we employed the Pearson correlation coefficient as the evaluation metric, which measures the linear correlation between the true and predicted similarity values.

5.3 Experimental details

All of our training used the $BERT_{\text{Base}}$ with provided pre-trained weights. For all experimentation, the learning rate was set to 2×10^{-5}, the hidden layer dropout probability was 0.3. All models were trained using an AdamW optimizer, which had the following parameters: $\beta_1 = 0.9$, $\beta_2 = 0.999$, and a weight decay regularization of $\lambda = 0.01$. We trained our model for 10 epochs using NVIDIA T4 GPUs for 3 tasks: SST Accuracy, QQP Accuracy, and STS Correlation.

5.4 Results

Table 1: Dev performance of different model architectures

Architecture	**SST Accuracy (Dev)**	**QQP Accuracy (Dev)**	**STS Correlation (Dev)**	**Overall Dev Score**
Bert Baseline (Cross-encoding)	0.305	0.665	0.169	0.518
Bert Baseline + Add. Layers	0.334	0.671	0.328	0.557
Bert Baseline + Add. Layers + MNRL	0.350	0.698	0.330	0.571
SBERT (Bi-encoding)	**0.336**	**0.738**	**0.684**	**0.639**

Our first experiment, compared different baseline model architectures. We tried 4 approaches: $BERT_{\text{Base}}$, $BERT_{\text{Base}}$ with additional layers, $BERT_{\text{Base}}$ with additional layers and MNRL, and finally SBERT. Based on the results of Table 1, adding additional hidden layers, greatly improved performance, especially on the semantic textual similarity task, bringing accuracy from 0.169 to 0.328. Moreover, the implementation of Multiple Negative Ranking Loss (MNRL) in addition to

the additional hidden layers and cross encoding, further improved all three tasks, with an overall dev score increasing from 0.557 to 0.571. However, the improvement was minimal, and after taking a closer look at our training results we noticed a lot of overfitting. To combat this issue we decided to implement SBERT, which improved the model greatly across all tasks as expected from Section 3. However, since STS and Paraphrase Detection are sentence-pair tasks, we wanted to see the effects bi-encoding and cross-encoding on the dev results as mentioned in section 4.3.

Table 2: Dev performance of different sentence encoding architectures with MNRL

Architecture	**SST Accuracy (Dev)**	**QQP Accuracy (Dev)**	**STS Correlation (Dev)**	**Overall Dev Score**
SBERT + Bi-enc. + MNRL	0.309	0.729	**0.662**	0.612
SBERT + Cross-enc. + MNRL	**0.311**	**0.779**	0.644	**0.615**

With our second experiment, we aimed to improve SBERT's performance across all three downstream tasks, particularly with QQP and STS because they are sentence-pair tasks. To do so, we started with 2 approaches: SBERT with bi-encoding for both QQP and STS and using MNRL, SBERT with cross-encoding for both QQP and STS, and using MNRL. We discovered that SBERT with MNRL achieves the highest QQP Dev Accuracy score with cross-encoding and the highest STS Dev Correlation with bi-encoding. This isn't what we expected as the SBERT baseline model performs better with bi-encoding for both sentence-pair tasks according to Reimers et al. (2019) and our data in Table 1. This indicates that SBERT cross-encoding captured the nuance between sentence pairs more accurately for QQP when combined with MNRL. This is likely because cross-encoding allows the model to jointly consider both sentences in a pair, leading to a more nuanced understanding and better identification of paraphrases, while bi-encoding might be more effective for STS due to its ability to handle semantic similarity through separate sentence representations. Thus, we transitioned to a model that used cross-encoding for QQP and bi-encoding for STS.

Table 3: Dev performance of mixed sentence encoding architectures and the affect of MNRL

Architecture	**SST Accuracy (Dev)**	**QQP Accuracy (Dev)**	**STS Correlation (Dev)**	**Overall Dev Score**
SBERT + Para. = Cross-enc. & Sim. = Bi-enc.	**0.358**	**0.772**	**0.660**	**0.653**
SBERT + Para. = Cross-enc. & Sim. = Bi-enc. + MNRL	0.322	0.765	0.653	0.638

With our third experiment, we aimed to determine the effect of MNRL on our model and if using cross-encoding for the QQP task and bi-encoding for the STS task made any improvements. We tried 2 approaches: SBERT with cross-encoding for QQP and bi-encoding for STS using MNRL and SBERT with cross-encoding for QQP and bi-encoding for STS without using MNRL. For the model including MNRL, the improvements weren't as drastic as we had expected with the SST Dev accuracy only improving by around 0.11, STS Dev Correlation decreasing by 0.009, and QQP decreasing by 0.014 as compared to the best results of Table 2. This is likely because the mixed encoding approach did not harmonize well with the MNRL's loss function, potentially leading to suboptimal gradient updates during training and, consequently, less effective feature learning for these tasks. This is further exemplified by the results of Table 3 where the model without MNRL outperformed the model with MNRL in every Dev set.

For our final experiment, we wanted to improve the SST accuracy so we implemented OLL on the SBERT model with cross-encoding for QQP and bi-encoding for STS without using MNRL. Our final model implementation achieved the following scores on the dev set and test set as displayed in Table 4. The following results were expected because OLL addresses the ordinal nature of the sentiment classification task by explicitly penalizing predictions that are farther from the true labels to better capture the nuances of sentiment analysis. This alignment with the ordinal structure of sentiment data contributed to more accurate predictions, improving the SST accuracy.

Table 4: Dev and Test performance of mixed sentence encoding architectures with OLL

Architecture	SST Accuracy	QQP Accuracy	STS Correlation	Overall Score
(Dev) SBERT + Para. = Cross-enc. & Sim. = Bi-enc. + OLL	0.385	0.772	0.660	0.662
(Test) SBERT + Para. = Cross-enc. & Sim. = Bi-enc. + OLL	0.397	0.772	0.624	0.660

6 Analysis

6.1 Sentiment Analysis

Despite integrating various extensions into our base BERT model, there was minimal improvement when considering the Sentiment Analysis task. However, these results may be attributable to the nature of each of the extensions that we integrated. Our largest improvements were seen through the implementation of the S-BERT-inspired architecture and the Multiple Negative Ranking Loss. The S-BERT-inspired architecture, which is inherently designed for tasks involving comparisons between sentence pairs due to its implementation of bi-encoding and cosine similarity fine-tuning, would not have made much of an improvement in the sentiment analysis task. MNRL, which is mainly aimed towards differentiating between examples, also did not perform well on the sentiment analysis task likely due to the absence of a comparative element within the task. Overall, these results demonstrated the need for specific extension and adaptation for the sentiment analysis task. As such, we aimed to implement Ordinal Log-Loss mainly to improve model performance on sentiment analysis, a task where the ordering of the sentiment classes has significant meaning. When analyzing the results of the Ordinal Log-Loss implementation, we saw moderate success in improving model performance on sentiment analysis.

6.2 Paraphrase Detection

Paraphrase detection was a task that our model improved fairly significantly, with our baseline model receiving an accuracy score of 0.558, and the best extension of our model receiving an accuracy score of 0.779. Implementing the S-BERT architecture, which focused on developing efficient processes for analyzing sentence pairs and utilized cosine similarity for fine-tuning, was highly effective in creating more robust embedding pairs. This method not only improved the accuracy of detecting paraphrases by focusing on semantic similarity but also enhanced the model's efficiency by allowing pre-computed embeddings to be reused in multiple comparisons. Furthermore, MNRL, which aims to optimize similarity between positive pairs while increasing the difference between negative pairs, significantly improved the development of pairwise sentence embeddings.

In evaluating our model's performance on paraphrase detection, we saw that the model had difficulty with sentence pairs that were highly similar in their words but differed in key details. These were often misclassified as paraphrases. For example, the questions "Is it better to bathe with cold water during winter or with hot water?" and "Is there a better time to drain your water heater? Can you do it during the winter?" were incorrectly identified as paraphrases. Despite their surface similarity in discussing water and winter conditions, these questions address fundamentally different topics. This and similar examples demonstrate difficulty in understanding more specific semantic differences within sentences paired with similar syntax. For future research, exploring advanced attention mechanisms could help to more effectively discern nuances, leading to more accurate paraphrase detection

6.3 Semantic Textual Similarity

Similar to the task of paraphrase detection, our model also improved significantly on the task of semantic textual similarity. Our model started with a Pearson correlation score of 0.169, indicating poor performance. As such, enhancing the baseline model was a key focus. By adding another hidden layer, cross-encoding with [SEP], and MNRL, we increased our score to 0.330. This improvement likely stems from the model's enhanced ability to capture nuanced similarities through the additional layer and more sophisticated encoding of sentence pairs. However, the most significant improvement for this task was the adjustment of our model architecture, which used bi-encoding and was inspired by the architecture implemented by S-BERT, which brought our score to 0.684. This is due to the

more sophisticated sentence embedding techniques, which work to better capture deep semantic relationships. This improvement in semantic textual similarity was expected when implementing this architecture, as the original authors had seen similar improvements for similar NLP tasks.

When reviewing incorrectly labeled examples for this task, we found that we had similar instances of model misclassification as in the paraphrase detection task. Sentences with similar structures but different meanings were often misclassified as similar, when they, in reality, were not. For example, "Russia, China Veto UN Resolution on Syria" and "Russia, China veto UN resolution on Syria killings" were misclassified to have more similar meaning, likely due to surface-level similarity, and incorrect due to the deeper difference in meaning between the two sentences.

7 Conclusion

In this paper, we presented an approach to enhance the BERT model's performance in multitask learning by incorporating various encoding strategies and specialized loss functions. Our main successes were observed in paraphrase detection and semantic textual similarity tasks, validating the effectiveness of S-BERT architectures, Multiple Negatives Ranking Loss (MNRL), and Ordinal Log-Loss (OLL) implementations. The application of bi-encoding for semantic textual similarity and cross-encoding for paraphrase detection optimized the model. Implementations of MNRL and S-BERT architectures improved sentence pair embeddings, while OLL enhanced sentiment analysis by adjusting the adjacent boundaries between labels. However, our work has limitations. MNRL was unable to successfully improve the model while utilizing mixed sentence encoding architectures. Moreover, our model overfit on the smaller SST and STS datasets. Lastly, dataset diversity was limited, suggesting future work should incorporate a broader range of data sources.

8 Ethics Statement

The development of our miniBERT model for NLP tasks raises several ethical issues and significant societal risks. One societal risk is the possibility of biased outputs due to imbalanced data representation. Our model isn't trained on a diverse set of datasets that represent a diverse set of groups. Specifically, we used three main datasets for our experiments: the Stanford Sentiment Treebank (SST) for sentiment classification, the Quora Question Pairs dataset for paraphrase classification, and the SemEval Semantic Textual Similarity (STS) Benchmark dataset for similarity classification. The SST dataset, with sentences from movie reviews, does not capture the full range of sentiment expressions across different cultures and languages. The Quora dataset, while extensive, primarily represents English-language questions, potentially missing nuances from other languages, cultures, and contexts. The SemEval STS dataset, despite its focus on semantic similarity, is limited in its cultural and linguistic diversity. Therefore, in the context of similarity analysis, sentiment analysis or paraphrase detection, this could lead to unfair or biased assessments of minority groups, languages, or dialects because the training data did not include sufficient representation of these groups, thereby reinforcing societal prejudices. Another ethical concern about the usage of NLP models like miniBERT is their potential for automating content moderation and censorship, which could disproportionately impact marginalized groups. For example, if a government or corporation uses miniBERT to monitor and filter online content, there is a risk that these models could misinterpret context or cultural nuances, leading to the unjust silencing of voices from marginalized communities. This can occur under the guise of maintaining public safety or upholding community standards, ultimately stifling free speech and exacerbating existing inequalities.

To address these issues, many mitigation measures can be implemented. To overcome data bias, we can use multiple datasets that cover a variety of linguistic, demographic, and cultural contexts. This entails not just gathering data from a variety of languages and dialects, but also ensuring that it reflects diverse socioeconomic, gender, and ethnic origins. You can also have multiple people interact and test the miniBert model, ensuring that is accessible and welcoming to a diverse audience. Furthermore, to mitigate issues of improper content moderation, we can make open-source datasets and architecture for peer evaluation to limit individual bias; however, we must make sure to not disclose the model's weights so that people can't take advantage of the model. We could also implement a method in which decisions by the model are explained by the model, to hopefully provide a clearer picture of the potential biases in the model.

9 References

[1] Jacob Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2018. URL: https://arxiv.org/abs/1810.04805 (visited on 01/01/2022).

[2] Matthew Henderson et al. *Efficient Natural Language Response Suggestion for Smart Reply*. 2017. URL: https://arxiv.org/abs/1705.00652 (visited on 01/01/2022).

[3] Richard Socher et al. "Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank". In: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2013.

[4] Nils Reimers and Iryna Gurevych. *Sentence-BERT: Sentence Embeddings Using Siamese BERT-Networks*. 2019. URL: https://arxiv.org/abs/1908.10084 (visited on 01/01/2022).

[5] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2014. URL: https://arxiv.org/abs/1412.6980 (visited on 01/01/2022).

[6] François Castagnos, Martin Mihelich, and Charles Dognin. "A Simple Log-Based Loss Function for Ordinal Text Classification". In: *Proceedings of the 29th International Conference on Computational Linguistics*. International Committee on Computational Linguistics. Gyeongju, Republic of Korea, 2022, pp. 4604–4609.

[7] Glanceable-io. "Ordinal Log Loss - A simple loss function for Ordinal Classification". In: (2023). URL: https://github.com/glanceable-io.

10 A Appendix

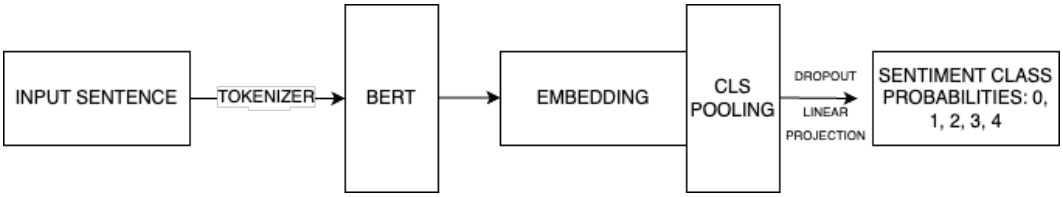

Figure 3: Sentiment Analysis Architecture

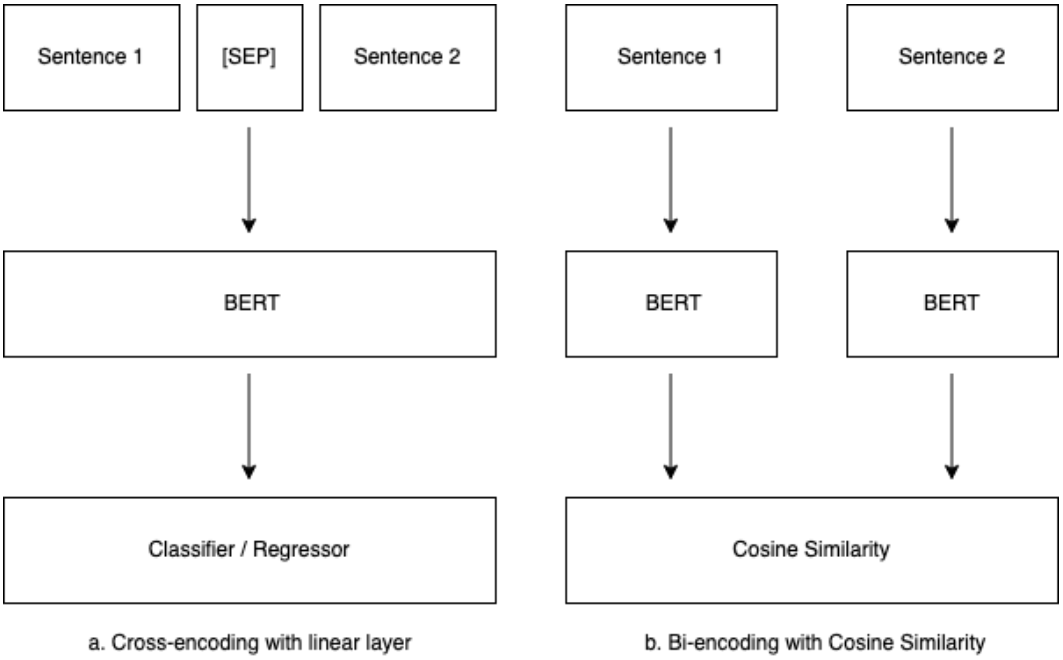

Figure 4: Encoding methods for multiple input sentences