MinBERT Task Prioritization, Cross-Attention and Other Extensions for Downstream Tasks

Stanford CS224N Default Project

Parker Stewart and Armando Borda
Department of Computer Science
Stanford University
`parkers@stanford.edu, aabd@stanford.edu`
Shared two late days from aabd

Abstract

This project involves implementing the core components of the minBERT model and extending its multitask classifier performance on the sentiment analysis, paraphrase detection, and semantic textual similarity tasks. Our work extends minBERT's downstream task applications through data prioritization, cross-attention mechanisms and the incorporation of other extensions like attention pooling and cosine similarity for similarity prediction. By integrating these enhancements, we seek to leverage multi-task learning. We found that implementing these multitask classifier extensions yielded improvements in the accuracy for the sentiment and especially the paraphrase task and a significant improvement in the Pearson correlation results for the semantic similarity task. This represents a step forward in utilizing transformer-based models for complex language understanding tasks with limited computational overhead.

- Mentor: Rohan Taori
- Team contributions: Parker worked on implementing the cross-attention implementation, attention pooling, cosine similarity, and hyperparameter tuning, and Armando worked on the task prioritization, hyperparameter tuning, and attention debugging.

1 Introduction

In the field of Natural Language Processing (NLP), the introduction of transformer-based models and constantly improving architectures has redefined the boundaries of how these language models interact in different problems. MinBERT, from the paper "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" (Devlin et al., 2019), introduces a very powerful framework designed for greater accessibility and efficiency across diverse computational settings. Moreover, it has the ability to, in a compact way, tackle different downstream tasks. This model represents a pivotal shift towards making advanced NLP capabilities available in environments where resource constraints have traditionally imposed limits on technological deployment. However, we considered there was still a lot of potential to explore the downstream tasks implementation process for this model to improve performance.

In this paper, we worked on implementing the fundamental architecture of minBERT, the pretraining process, and the finetuning phase. Regarding the last one, we focused on the following tasks: sentiment analysis, paraphrase detection, and semantic textual similarity. Most importantly, we focused on extending the functionality on the additional downstream tasks by introducing extensions such as task prioritization through weighting, inspired by ideas from the paper Dynamic Task Prioritization for Multitask Learning (Guo et al., 2018). We also introduced an approach regarding cross-attention and attention pooling inspired from the paper Cross-Attention is All You Need: Adapting Pretrained Transformers for Machine Translation (Gheini et al., 2021). With this, our aim was to use multi-task learning by allowing our model to adapt to the different tasks with a new framework.

2 Related Work

The introduction of BERT ("Bidirectional Encoder Representations from Transformers") (Devlin et al., 2019) represents a significant advancement in the development of language models. BERT's approach is novel given that it does pre-training through bidirectional representations, hence conditioning on both left and right contexts across all layers. This approach enables the fine-tuning of the pre-trained model with limited output layers to do very well in a wide array of NLP tasks, without significant task-specific modifications. BERT's use of a masked language model (MLM) for pre-training allows it to integrate context from both directions, setting it apart from unidirectional models and achieving state-of-the-art results across various benchmarks. This innovation, which is used by trying to predict the original words now masked based on its context, not only underscores the importance of bidirectional pre-training but also simplifies the architecture for downstream tasks, demonstrating BERT's high effectiveness.

Furthermore, there has been significant interest in enhancing the adaptability of models to multiple tasks. The concept of multi-task learning in the context of transformers involves fine-tuning a pre-trained model across different tasks simultaneously. Despite its potential, multi-task learning with transformers presents challenges, particularly in trying to maintain computational costs low.

To address these challenges, the paper Dynamic Task Prioritization for Multitask Learning (Devlin et al., 2019) proposed a method to prioritize difficult tasks during training, as opposed to curriculum learning where easier tasks are prioritized. The key for this was to understand that each task weight is used to combine multiple loss objective functions. Then, the way the model identifies harder tasks is by looking at their key performance indicators. This papers also discusses about hard parameter sharing, where a hidden layer is shared across many tasks to achieve multitask learning, but its main drawback is the requirement to combine task specific loss objectives, which requires task-specific weights that are expensive. The paper concludes that dynamic task prioritization achieved similar levels of performance compared to single-task models.

We were also inspired by the work in Cross-Attention is All You Need: Adapting Pretrained Transformers for Machine Translation (Gheini et al., 2021), which explores the high effectiveness of using cross-attention for machine translation. The research paper focused on using cross-attention to potentiate the efficacy of fine-tuning the cross-attention layers in a transformer-based machine translation model. It involved fine-tuning a pre-existing translation model to adapt to new language pairs, either by changing the source or the target language. Moreover, the paper concludes that fine-tuning solely the cross-attention layers nearly matches the performance achieved by fine-tuning the entire model. This also reduces the need for high parameter storage overhead when extending machine translation models to additional language pairs. Additionally, this paper highlights the underappreciated power of cross-attention in transfer learning contexts, suggesting a more parameter-efficient fine-tuning method that can be used to extend some other downstream tasks. The adaptability of this approach to different language pairs will be instrumental as we expand its application to tasks involving semantic similarities and paraphrase detection.

3 Approach

Our approach started off with implementing the core minBERT functionality. We implemented the calculation of attention scores, the add-norm function that applies a dense layer and dropout to the layer's output, the multi-head self-attention mechanism in the forward function and the processing of input and position embeddings. We then implemented the core functionality of the classifier module, with the main classifier being a sequential model composed of a linear layer, activation function, and another linear layer. In the later sections, we focused our approach in improving the downstream tasks.

3.1 Basic Multitask Classifier

Our first implementation, which we call Basic Multitask Classifier, consists of four methods: forward(input_ids, attention_masks), predict_sentiment(input_ids, attention_masks), predict_paraphrase(input_ids, attention_masks), and predict_similarity(input_ids, attention_masks). At a high level, the goal of the forward layer is to utilize the minBERT model to gain information about the sentences, which it then passes on to the task specific heads, which make the predictions for their respective tasks.

The 'forward' method extracts minBERT embeddings from input sentences (input_ids), by passing the input sentences and their respective attention masks through the BERT model. It then takes the [CLS] token's pooled outputs, summarizing the sentence's context and applies a dropout of 0.3 to these outputs to combat overfitting before returning them for use in downstream tasks.

'predict_sentiment' outputs logits for sentiment classification into five categories, from negative to positive, based on sentence embeddings from the 'forward' method. These embeddings are linearly mapped to sentiment logits by the 'sentiment_prediction' layer.

'predict_paraphrase' determines if sentence pairs are paraphrases, using the sentence's pooled embeddings to form a difference vector for semantic dissimilarity. This vector feeds into the linear 'paraphrase_prediction' layer, producing a paraphrase likelihood logit.

'predict_similarity' assesses semantic textual similarity between sentence pairs, constructing a difference vector from pooled embeddings to highlight semantic differences. This vector is passed through the linear 'similarity' layer, yielding a logit that reflects the sentences' semantic similarity. This model's results will be used as our implementation of a baseline.

3.2 Task Prioritization and Cross-Attention Multitask Classifier

We extended the previous multitask classifier to our Task Prioritization and Cross-Attention Multitask Classifier. It kept part of the same structure as the Basic Multitask Classifier, with a forward function that takes in input sentences with attention masks, extracts key information vectors from the sentences, and passes those vectors to the task specific heads for downstream tasks. In the cross-attention mechanism part of this improvement, we extended the use of cross attention explained in the Cross-Attention paper (Gheini et al., 2021), which focused on token interaction with minimal parameter updates, by selectively updating cross-attention layers to amplify the semantic signals for accurate paraphrase detection and semantic textual similarity.

- **Forward Function:**
  - *Sentiment Classification:* Processes single sentences to extract BERT's pooled output embeddings for sentiment classification. The pooled embeddings, primarily from the [CLS] token, are represented as:

    $$PooledOutput_{sentiment} = BERT_{pooler}(input\_ids, attention\_mask) \quad (1)$$

  - *Paraphrase Detection and STS:* For sentence pairs, employs cross-attention and attention pooling to capture inter-sentence relationships:
    * *Cross-Attention Mechanism:* Applies bidirectional attention between tokens of two sentences, formulated as:

      $$Attention(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = softmax\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V} \quad (2)$$

      where queries $\mathbf{Q}$ and keys/values $\mathbf{K}, \mathbf{V}$ are derived from token embeddings $\mathbf{H}_1, \mathbf{H}_2$ of the sentence pair.
    * *Attention Pooling Mechanism:* Aggregates cross-attended embeddings into single vectors for each sentence, summarizing enriched information via:

      $$Pooled_i = \sum_j \alpha_{ij}\mathbf{h}_j \quad (3)$$

      using attention weights α_{ij} and cross-attended embeddings $\mathbf{h}_j$.

  * The pooled vectors from attention pooling are returned for downstream tasks, encapsulating the relational dynamics essential for paraphrase detection and STS evaluation.
  * Another way to view this implementation is through the pair of sentence embeddings in the paraphrase or sentiment tasks E_{CLS1} and E_{CLS2}:
    $$\mathrm{A}_1, A_2 = CrossAttention(Dropout(E_{CLS1}), Dropout(E_{CLS2}))$$
    $$\mathrm{E}_{combined} = ReLU\left(Dropout\left(SharedLayer2\left(Concat(A_1, A_2)\right)\right)\right)$$
    The general objective for learning can be formulated as minimizing the loss L, between the predicted outputs of the model $f(E_{combined})$ and the true labels y, across all training examples:
    $$\mathrm{O} = \min_\theta \sum L\left(f(E_{combined}; \theta), y\right)$$
    where θ represents the parameters of the entire model, including minBERT, the cross-attention layers, and the additional shared layers. This objective encapsulates the model's learning process. This is further explained below.

- **Predict Sentiment:**
  - Uses the pooled output from the 'forward' function for single sentences. After applying dropout, the pooled output is passed through a ReLU function ($ReLU(x) = \max(0, x)$) to introduce non-linearity for enhanced feature representations. The 'sentiment_prediction' linear layer then maps these processed embeddings to logits for five sentiment classes.
- **Predict Paraphrase:**
  - For sentence pairs, this method utilizes the pooled vectors obtained from the 'forward' function. A vector difference between the pooled embeddings ($\mathbf{pooled\_1} - \mathbf{pooled\_2}$) is computed and concatenated with pooled_1 and pooled_2. The concatenated vector is then passed through a normalization layer, followed a ReLU activation to introduce non-linearity, which is passed to dropout. The resulting vector is then input to the 'paraphrase_prediction' layer, producing a logit that reflects the likelihood of the sentences being paraphrases.
- **Predict Similarity:**
  - For computing semantic textual similarity between sentence pairs, the method first calculates the cosine similarity between the pooled embeddings from the 'forward' function:
    $$CosineSimilarity = \frac{\mathbf{pooled\_1} \cdot \mathbf{pooled\_2}}{\|\mathbf{pooled\_1}\|\|\mathbf{pooled\_2}\|} \tag{4}$$
    After applying dropout to enhance generalization, the similarity score is normalized from $[-1, 1]$ to a $[0, 5]$ scale to conform with typical STS benchmarks. This score quantifies the semantic similarity between the sentence pairs.
- **Training Function:** Manages the training process of the MultitaskBERT model, dynamically adjusting task focus to mitigate overfitting and improve generalization across tasks.
  - *Loss Functions:*
    * *Sentiment Classification:* Uses Cross-Entropy Loss to compare the output logits against true class labels.
      $$Loss_{sentiment} = -\sum_{c=1}^{M} y_{o,c} \log(p_{o,c}) \tag{5}$$
      where M is the number of classes, y is a binary indicator of class label c, and p is the predicted probability of class c.
    * *Paraphrase Detection:* Employs Binary Cross-Entropy Loss for the prediction of paraphrase likelihood.
      $$Loss_{paraphrase} = -\left[y \log(p) + (1 - y) \log(1 - p)\right] \tag{6}$$
      where y is the true label (0 or 1) and p is the predicted probability of being a paraphrase.

    * *Semantic Textual Similarity:* Utilizes Mean Squared Error Loss to capture the deviation from true similarity scores.

$$Loss_{similarity} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{7}$$

      where y_i is the true similarity score, and $\hat{y}_i$ is the predicted score.
  - *Dynamic Training Scheduler:* Adjusts the likelihood of selecting each task during batch processing based on performance metrics to reduce overfitting. We started by inspiring our methodology from the prioritization in Dynamic Task Prioritization for Multitask Learning (Guo et al. (2018)), but still with a distinct approach.
    * Each task is initially assigned a selection weight. These weights are dynamically adjusted based on the relative performance of each task, promoting balance in training focus and reducing the risk of overfitting on any single task.
    * Weight Calculation for Task Selection is roughly:

$$Weight_{task} = \frac{test\_acc_{task} - dev\_acc_{task}}{\sum_{all} test\_acc_{task} - dev\_acc_{task}} \tag{8}$$

      We also incorporated a smoothing factor adjustment α which linearly decreases from an initial value to a final value. This is used to be multiplied with the proposed weight added to (1-α) multiplied by the original weight, giving us new weight values to use. We finally normalize to 1.
    * This dynamic scheduling approach helps in evenly distributing the learning focus across tasks, especially when dealing with datasets with drastically different sizes, thereby enhancing the model's overall generalization ability.

When training this model, we saw overfitting issues, where the training section performed much better than the dev portion, so this had to be adjusted iteratively by stopping the training process and performing hyperparameter tuning, among other changes.

4 Experiments

4.1 Data

- **Sentiment Analysis**:
  - The Stanford Sentiment Treebank, consisting of 11,855 single sentences from movie reviews. The labels for the reviews range from 0-4, labeling their sentiment.
  - The CFIMBD Dataset, consisting of 2,434 polar, binary movie reviews.
- **Paraphrase Detection**:
  - The Quora Dataset, consisting of 400,000 question pairs with binary labels for paraphrases.
- **Semantic Textual Similarity**:
  - The SemEval STS Benchmark Dataset, consisting of 8,628 sentence pairs with classification labels from 0-5 indicating how similar the sentences are.

4.2 Evaluation method

We evaluated Sentiment Analysis and Paraphrase Detection with accuracy, the ratio of correct predictions to the total amount of data. We evaluated Semantic Textual Similarity with the Pearson correlation of the true similarity values against the predicted similarity values.

4.3 Experimental details

For Sentiment Analysis, we trained minBERT on the Stanford Sentiment Treebank and the CFIMBD Dataset. For the baseline multitask classifier, minBERT was trained on the STS, Quora Dataset, and the SemEval STS Benchmark Dataset.

For all pretraining runs, we used a learning rate of 1E-5. For finetuning, we used a learning rate of 1E-3.

We used 10 epochs for all pretraining and finetuning runs in every model.

For the Task Prioritization and Cross-Attention Multitask Classifier, we set the max learning rate in the dynamically adjusted setting to 0.001, and the starting learning rate to be the one specified as the argument when running the model.

4.4 Results

Table 1: Multitask Classifier Performance

Model	**Overall Score**	**Sentiment Acc**	**Paraphrase Acc**	**Similarity Corr**
Basic Multitask	0.406	0.468	0.609	0.142
Extended	0.719	0.495	0.845	0.632

The Extended Classifier above is the Task Prioritization and Cross-Attention Multitask Classifier. Our first general observation is that as expected, the advanced model's results are considerably higher than the baseline results. We obtained much higher accuracy and Pearson correlation results with the Task Prioritization and Cross-Attention Multitask Classifier compared to the Basic Multitask Classifier.

From a relative perspective, we had a significant increase from our baseline. However, within the Task Prioritization and Cross-Attention Multitask Classifier results, we saw that improving the Multitask Classifier after we had introduced most modifications and we tried to reparameterize it to get better results was challenging. The biggest increase was with the similarity correlation, with a 345 percent increase. The paraphrase accuracy increased by 28 percent. The sentiment accuracy increased by around 6 percent.

5 Analysis

The introduction of task prioritization and cross-attention mechanisms aimed to refine the model's focus and enhance its ability to differentiate semantic differences and similarities across tasks. These additional layers of complexity translated to performance gains. The improvements suggest that the model may have encountered increasing returns from these complexities, where the added model sophistication meant higher task performance. This is especially true with the use of cross-attention in the paraphrase and similarity tasks.

The dynamic task prioritization was designed to adjust the model's focus based on performance metrics, potentially improving learning across tasks. This strategy's effectiveness is inherently tied to the diversity and distribution of the training data. Hence, even though we saw improvements in our results with the Task Prioritization and Cross-Attention Multitask Classifier, it is reasonable we didn't get near perfect scores. Moreover, the introduction of the smoothing factor α, explained in the Approach section, was necessary as the updates to the weights were too abrupt and the model too unstable when this was removed. Hence, the inclusion of α was crucial and we kept it in the model.

When running the Task Prioritization and Cross-Attention Multitask Classifier, we ran into significant overfitting issues. The model was ran repeatedly to a few epochs and when this issue arose, training was cancelled and modifications were made. We analyzed the data and obtained around 0.98 in the training results for semantic similarity, for instance, but a dev result around 0.60. This shows an extremely large amount of overfitting. Hence, a lot of inspection was done in modifying the task heads, adding and switching complexities, in order to reduce this issue.

With regards to the cross-attention, if the subsequent layers or task-specific classifiers are not finely adjusted to leverage the nuanced embeddings produced by cross-attention, the overall task performance might not reflect the expected improvements. There was probably a necessity for improved comprehensive model optimization, where there is a need to adjust the entire pipeline. This entails refining the embeddings, the cross-attention module, and especially the task-specific heads to ensure they are all able to exploit the sophisticated features generated. This could have let to even higher results. However, the improvement in the similarity task is impressive, and the absolute accuracy obtained for the paraphrase task is very high, which can be explained by the cross-attention introduction and not just taking a simple difference calculation like with the Basic Model.

With regards to the Pearson correlation, and compared to the basic model, we saw that the inclusion of cosine similarity for predict similarity also had a positive effect, which was added later into the model. Through the ablation analysis by comparing its absence with its inclusion in the Task Prioritization and Cross-Attention Multitask Classifier, we consider that due to the comparison in the directionality of embedding vectors, cosine similarity offered a direct measure of semantic alignment between sentences, which is less susceptible to the absolute differences that might mislead other forms of similarity assessment.

Furthermore, the model's performance on paraphrase detection and semantic textual similarity tasks was influenced by how effectively attention pooling managed to capture and summarize relevant semantic cues from the input sentences. By focusing on key tokens that bear the most relevance to the task at hand, attention pooling enabled the model to form a richer, context-aware embedding of each sentence. This was evidenced by the model's improved ability to distinguish between paraphrases and non-paraphrases, as well as to accurately evaluate the semantic similarity.

The task weighting complexity added to the model didn't help the sentiment accuracy a lot, which fell behind compared to the other tasks improvements. The increased focused given to this task through the task prioritization should have been complemented with a different processing strategy for this task. This is shown by the low increase in performance.

6 Conclusion

From this project, our main findings highlight the versatility and power of minBERT and its derivatives for addressing multiple NLP tasks within a single framework. We saw that it is adaptable to multitask learning, enhanced by the capacity to use cross attention mechanisms and its ability to use dynamic task prioritization. With regards to this, our approach to adjusting task priorities based on performance helped to balance learning across tasks of varying difficulty and improve the finetuning results. At first, the results from incorporating a cross-attention and task prioritization approach produced significant improvements, but even better once we incorporated attention pooling and cosine similarities.

A promising direction for future work is the investigation of more complex attention mechanisms beyond the standard cross-attention used in this work. For instance, transformer models incorporating adaptive or sparse attention patterns could provide a more nuanced understanding of sentence relationships, especially in tasks requiring high levels of semantic understanding like paraphrase detection and semantic textual similarity. Moreover, extending the model to encompass a broader range of NLP downstream tasks, such as question answering (QA), would allow us to test if the flexibility and scalability of the minBERT architecture can expand even more and contribute valuable insights into the general adaptability of transformer-based models in multi-task learning environments. Additionally, exploring sentiment-specific embeddings or attention mechanisms that can capture and utilize the emotional valence of texts could significantly enhance sentiment analysis accuracy. Through these efforts, we aim to push the boundaries of what is possible with current NLP technologies and pave the way for better, more effective language models.

References

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

Mozhdeh Gheini, Xiang Ren, and Jonathan May. 2021. Cross-attention is all you need: Adapting pretrained transformers for machine translation. Online.

Michelle Guo, Albert Haque, De-An Huang, Serena Yeung, and Li Fei-Fei. 2018. Dynamic task prioritization for multitask learning. Online.