Fine BERT

Stanford CS224N Default Project

Xavier Millan
Department of Computer Science
Stanford University
xavierm@stanford.edu

Yuvraj Baheti
Department of Computer Science
Stanford University
yuvrajb@stanford.edu

Andrew Nguyen
Department of Computer Science
Stanford University
andrewkn@stanford.edu

Abstract

Using a pretrained BERT (Bidirectional Encoder Representations from Transformers) model (Devlin et al, 2018), we sought to finetune for the tasks of sentiment analysis, semantic textual similarity, and paraphrase detection. These tasks are common natural language proccesing benchmarks which aim to evaluate a particular model's ability to understand the meaning and nuance of language. As a pretrained transformer based model, BERT already has an understanding of language, but isn't trained to properly complete these tasks; we therefore wanted to show in what way finetuning could enable it to do so best. Specifically, we implemented a wide number of different finetuning techniques and methodologies in a comparative manner to observe which methodologies performed best for any given task(s) and analyze the causes for such behavior. Through this research, we have determined that the finetuning methods which obtained the best results were different - in some cases very different - from task to task. In our paper, we will share our iterative process, provide some quantitative findings, and reason about our results.

1 Key Information to include

- Mentor: Aditya Agarwal
- External Collaborators (if you have any): None
- Sharing project: No
- Member Contributions: Andrew, Xavier, and Yuvraj distributed work equally and worked on every aspect of the project together.

2 Introduction

It is often said that language is what separates man from animals. Language is a medium through which we can express our ideas and our emotions. Language is how we encapsulate and codify the world around us. Language is powerful, beautiful, and complicated. A word can have many definitions - a glance at the dictionary entry for most words would prove this point - but any one of these definitions may itself have different flavors depending on the context in which the word was used: who said it, how did they say it, why did they say it, to who did they say it, ... In this way, no two usages of the exact same word will ever result in the exact same meaning for said word, and yet language so elegantly captures all of these millions of nuances, semantics, and connotations within the single word regardless.

In many ways then, designing a way for an algorithm, a computer, or any non-human entity to truly process, understand, and replicate natural language in spite of its many complexities might at first appear to be an impossibility. And yet for the past several decades, humans - or at least a very small subset of humans - have been working hard to do just that. In recent years this work has accelerated tremendously and culminated in the introduction of NLP models that seem to get increasingly closer to human-level performance on a variety of language processing tasks (Han et al, 2021).

Three common tasks in this vein are semantic textual similarity (STS), sentiment analysis, and paraphrase detection. The STS task evaluates a model's ability to rate the similarity of two input sequences on a continuous scale from 0 -5, where 0 represents no similarity in meaning and 5 represents virtually identical meaning (Cer et al, 2017). Sentiment analysis is a classification task, where for any given input, the model is asked to determine how positive or negative of sentiment is expressed implicitly in the input on a discrete scale from 0 - 5, where 0 represents very negative sentiment and 5 represents very positive sentiment (Medhat et al, 2014). Finally, in the paraphrase detection task, a model is given two input sequences and asked to make a binary decision on whether they are paraphrases of each other (Dolan et al, 2005).

Each of these tasks tries to evaluate one aspect of a model's ability to deeply understand language beyond individual word meanings, and in combination, a model's ability to perform across these tasks may serve as a proxy for evaluating its ability to process language in general.

As a primer, pretrained BERT, which we use as a baseline model, has been trained through unsupervised learning, where corpora of texts are used as input and by predicting the next word in any given text and using the generated loss thereof to update embeddings representing word meaning, the model learns about language in general (Devlin et al, 2018). Now, to accomplish these tasks in particular, we turn to supervised learning, using labeled data and methodologies elucidated below to train (finetune) BERT (Ding et al, 2023). Using different techniques for each task - namely a CNN convolutional neural network final layer for STS, a bidirectional LSTM (long short-term memory network) for paraphrase detection, and a simple linear layer with nonlinear activation for sentiment analysis - and experimenting with varying hyperparameters (learning rate, dropout probability, embedding representations, etc.), we found that a different combination of methods worked best for each of the three tasks and intend to share our process, results, and reasoning in the following sections.

3 Related Work

Research in the field of natural language processing in general and task-specific finetuning for semantic textual similarity, sentiment analysis and paraphrase detection in particular is broad and highly active. Many of the iterations of our model were based upon existing research that we used as guidance in our own implementation. Broadly speaking, research in optimizing model performance for these three NLP tasks falls into three categories: model selection and specification (the choice of model and the specifics of its implementation), endogenous optimizations (the selection of ideal hyperparameters and other model characteristics) and exogenous optimizations (choices made outside of the model itself that improve performance). The comprehensive list of topics for any of these categories is beyond the scope of this paper; below we simply enumerate the particular topics that we incorporated in one or more iterations of our model.

Model Selection and Specification
While BERT itself is a transformer based model, finetuning often involves the addition of layer(s) built atop the BERT architecture. A significant amount of research is based upon what model these layer(s) should be. Relevant to our own implementations, examples of such models are 1D Convolutional Layers (Kim, 2014), unidirectional and bidirectional Long Short Term Memory layers (Yin et al, 2017), and sequences of linear layers followed by non-linearities (Szandała, 2020). For classification tasks such as sentiment analysis, there exists research on various clustering techniques which may be implemented, namely k-means clustering and spectral clustering (Ren et al, 2022). Finally, the way in which the model should be trained - ie. the loss function which it is attempting to optimize - is also the topic of substantial research (Janocha et Czarneck, 2017).

Endogenous Optimization
For any particular model, there are additional optimizations which may be made to its specification. In this area, there exists extensive research on choosing ideal hyperparameters such as dropout probability and learning rate. There also exists research regarding the best way to represent differences

in embeddings for paired input situations, as for the parpharase detection and semantic similarity tasks (Mao et al, 2021). Finally, there are discussions regarding how (if at all) finetuning should change the embeddings of the base model - BERT in our case (Huo et Iwaihara, 2020).

Exogenous Optimizations
Outside of the model choice and optimization, there are also optimizations which can lead to enhanced performance. Namely, research has been done regarding the amount of data which a model is trained upon and the way in which it might be preprocessed for best results (Fan et al, 2021).

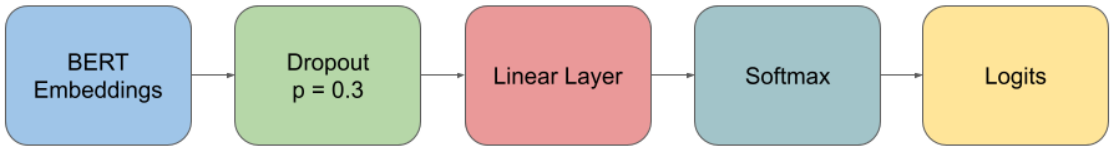
4 Approach

Given the lengthy number of iterations we implemented and the many approaches we tried, we will simply detail the approach with which we were able to achieve the best results here. We will elaborate on our entire iterative process - and all of the approaches thereof - in sections 5 and 6 below. In the end, the optimal approach consisted of training three task-specific models, one for each task.

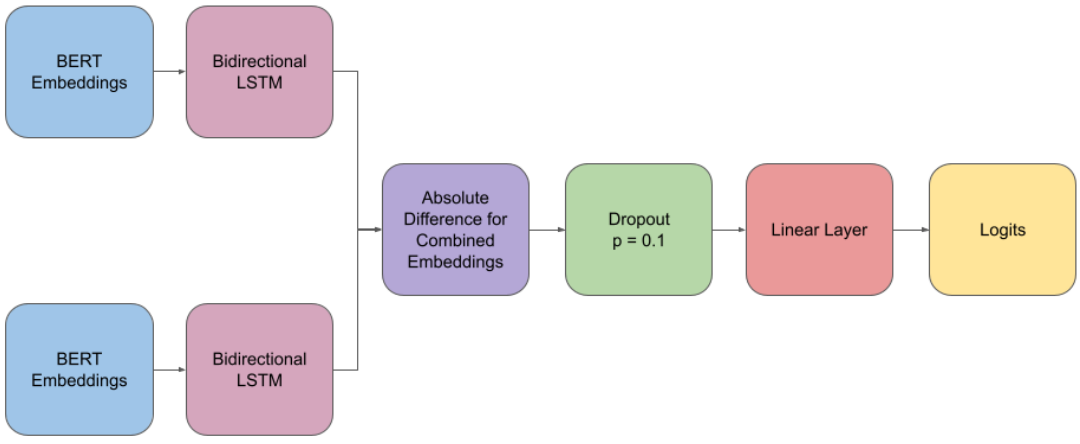
Sentiment Analysis: For this task, we found that implementing a simple sequence of linear layer and softmax activation (Eq. 1) performed best when combined with cross entropy loss (Eq. 2), dropout probability 0.3 and trained over 10 epochs on the full model at learning rate 1e-5 (Mao et al, 2023).

$$softmax(k, x_1, ..., x_n) = \frac{e_k^x}{\sum_{i=1}^{n} e_i^x} \tag{1}$$

$$\mathcal{L} = -\sum_{c=1}^{N} y_{o,c} log(p_{o,c}) \tag{2}$$

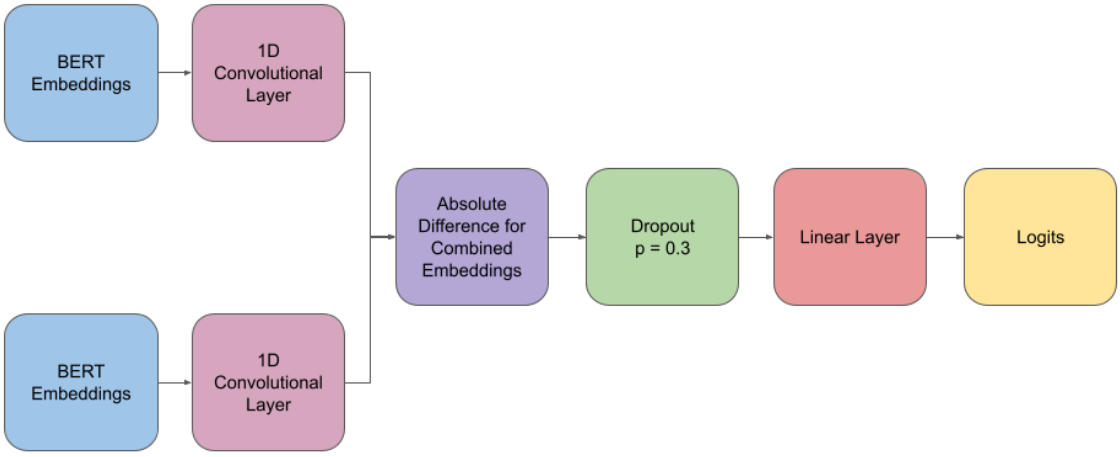


Paraphrase Detection: For this task, we implemented a sequence consisting of a bidirectional LSTM layer followed by a linear layer. We combined this with dropout probability 0.1 on the combined embeddings, which we obtained by taking the absolute difference of the paired embeddings generated by BERT and this final LSTM layer. We obtained best results when trained on only these last layers with learning rate 1e-3 over 10 epochs (Hochreiter et Schmidhuber, 1997).



Semantic Textual Similarity: For this task, we implemented a sequence consisting of a 1D Convolutional layer followed by a linear layer. We combined this with dropout probability 0.3 on the combined embeddings, which we obtained by taking the absolute difference of the paired embeddings generated by BERT and this final CNN (convolutional neural network) layer. We obtained the best

results when we trained on only these last layers with learning rate 1e-2 over 10 epochs (Kim, 2014).



5 Experiments

5.1 Data

Stanford Sentiment Treebank (SST): 215,154 unique phrases used for training and evaluation of sentiment analysis task, with discrete labels 1 - 5.
CMU Multimodal Opinion Sentiment and Emotion Intensity (MOSEI): 23,453 sentences with discrete labels 1 - 5 used for additional finetuning for sentiment analysis task.
Quora dataset: 404,298 question pairs with binary labels used for training and evaluation of paraphrase detection task.
SemEval STS Benchmark: 8,628 sentence pairs with continuous labels from 0 to 5 used for training and evaluation of semantic textual similarity task.
SemEval-2014 Sentences Involving Compositional Knowledge (SICK) dataset: 7,500 sentence pairs with continuous labels from 0 - 5 used for additional finetuning for semantic textual similarity task.

5.2 Evaluation method

We will use accuracy as the evaluation metric for paraphrase detection and sentiment analysis. Specifically, we will calculate the proportion of correct classifications that the trained model returns during evaluation for these two tasks. We will use Pearson correlation (Benesty et al, 2008) as the evaluation metric for semantic textual similarity.

5.3 Experimental details

Below, we provide brief specifications of the models we trained at each iteration. For brevity, any details which are not explicitly elucidated for an iteration should be assumed to be the same as the previous iteration. For example, if training rate for iteration *k* is not explicitly mentioned, it is the same as those of iteration *k - 1*. Many of the iterations are based upon existing work; where appropriate we include citation(s) to research we were inspired by in our own implementation at each iteration. We also include relevant functions, although for full details please refer to cited works.

Baseline Model: As a baseline model, we trained BERT for 10 epochs on the three standard datasets with learning rate 1e-5 for the last linear layer only, ie. the final layer that we implemented for each of the three tasks. We utilized a singular linear layer for each task and cross entropy loss as the objective function (Mao et al, 2023).

Iteration 1: We revised our STS task-specific finetuning methodology by removing an activation layer that previously restricted all logit outputs to a range of 0-1, which was incorrectly mapped to the label values of range 0 - 5 (Szandała, 2020).

Iteration 2: As opposed to training a single model on all three tasks, we implemented code to train three models - one per task. In this way, each model could be finetuned to only one task, both

improving performance and introducing task-specific training and improvement isolation. Each model was trained for 10 epochs on its respective standard dataset, with learning rate 1e-3 on the last linear layer only. We kept the same last layer and loss function for each task.

Iteration 3: With this new implementation, we could incrementally improve individual tasks without any impact on the performance of other tasks. We began by implementing *k*-means clustering (Eq. 3) as an additional layer for sentiment analysis. Specifically, using $k = 5$ for each of the classes, we attempted to assign each of the sentence embeddings in a particular batch to one of *k* groups before integrating this classification into the sentence embeddings for input to the final linear layer and softmax sequence. We trained the SST model alone, on the last linear layers with learning rate 1e-3 over 10 epochs (Krishna and Murty, 1999).

$$\mathcal{L} = \sum_{j=1}^{k} \sum_{i=1}^{n} \mathbf{1}(y^{(i)} = j) \|x^{(i)} - \mu^{(i)}\|^2 \tag{3}$$

Iteration 4: Disappointed in the results, we theorized that *k*-means may be limiting due to its assumption of convex clusters, which may not be true for the sentence embeddings. We thus implemented spectral clustering (Eq. 4) instead, similarly defining $k = 5$ for each class and using the learned classifications as part of the sentence embeddings for input to the final linear layer and softmax sequence. We additionally utilized low rank approximcation for embeddings as a method of noise reduction. Again, we trained the SST model alone, on the last linear layer with learning rate 1e-3 over 10 epochs (Ng et al, 2001).

$$\mathcal{L}(\alpha) = \frac{1}{N} \sum\nolimits_{n} F(W_n(\alpha), \prod\nolimits_{0}(e_n, \alpha)) + C\|\alpha\|_1 \tag{4}$$

Iteration 5: In light of the underperformance of clustering methodologies for sentiment analysis, we removed them from our implementation and shifted our attention to the STS task. Specifically, we focused on the way in which the paired embeddings were combined. Instead of concatenating them as previous, we took the absolute difference of the embeddings, which better represented the similarity/dissimilarity in the paired sentences. We trained on the standard datasets with learning rate 1e-3 on the last linear layer over 10 epochs. (Mao et al, 2021).

Iteration 6: Continuing with STS-specific optimization, we combined the standard STS Eval dataset with the STS SICK dataset and trained on this extended data. Due to potential differences in scoring/labeling of the data however, we actually observed a performance decrease in spite of additional data when training the STS-specific model for 10 epochs at learning rate 1e-3 for the last linear layer (Lin et al, 2022).

Iteration 7: We pivoted to implementing cosine similarity (Eq. 5) as an optimization for the STS task, calculating the cosine similarity of the two embeddings generated by the paired sentences and integrating these cosine similarity scores into the combined embedding before passing it through the last linear layer. We trained on the standard dataset alone for 10 epochs at learning rate 1e-3 for the last linear layer (Chunjie et al, 2017).

$$cosine - similarity = \frac{\sum_{i=1}^{n} \mathbf{A}_i \mathbf{B}_i}{\sqrt{\sum_{i=1}^{n} \mathbf{A}_i^2} \sqrt{\sum_{i=1}^{n} \mathbf{B}_i^2}} \tag{5}$$

Iteration 8: As opposed to using a linear layer / series of linear layers followed by a non-linear activation(s) for the final layer of all three task-specific models, we added a bidirectional LSTM (long short term memory) layer for all 3 tasks, based on the premise that this architecture could better encapsulate and learn relationships beyond the capability of a simple linear layer. We trained all three models for 10 epochs at learning rate 1e-3 on the last layer - this newly implemented LSTM layer (Hochreiter and Schmidhuber, 1997).

Iteration 9: Noting the improved performance this LSTM layer afforded, we added addition linear layers and interspaced non-linear activations (ReLU and/or Sigmoid) of varying dimensionality sequences to each of the models. Even after extensive trial and error, our best results obtained with these additional linear layers (detailed below) was lower than with simply one linear layer following the LSTM layer, as in the original implementation from Iteration 8. Results recorded were obtained from training all 3 models over 10 epochs at learning rate 1e-3 on the last layer(s) (Szandała, 2020).

Iteration 10: Motivated by the success of adding this LSTM layer, we replaced it with a 1D Convolutional layer instead, in the hopes that this architecture could better encapsulate meaning and nuance through its focus on neighboring words as a basis for word embeddings. For each model, the finetuning layers now consisted of a 1D CNN layer, followed by a linear layer. We trained all 3 models over 10 epochs at learning rate 1e-3 on the last layer(s) (Kim, 2014).

Iteration 11: Focusing once more on improving STS-task performance, we implemented preprocessing of the SemEval dataset by stripping any punctuation and converting all letter to lowercase prior to training. We then trained over 10 epochs on the last layers at learning rate 1e-3 (Fan et al, 2021).

Iteration 12: Iteration 11's failure to deliver better results was due to BERT's pretraining, which was done on properly formatted data. In order for this preprocessing to improve performance, we needed to train on the full model and allow BERT specific layers to be updated during training as well. We also changed dropout probability from the 0.1 we utilized for last linear layer training to 0.3 for the full model training. After training for 10 epochs on the full model at learning rate 1e-5, however, the observed results remained undesirable for all tasks except sentiment analysis (Devlin et al, 2018).

Iteration 13: Realizing that different tasks may benefit from full model training while others did not, we adopted a hybrid training approach, where the sentiment model was trained on the full model at learning rate 1e-5 over 10 epochs while the remaining two tasks were trained on the last layers only, at learning rate 1e-3 over 10 epochs.

Iteration 14: As a second attempt at full model training, we implemented gradual layer updating, wherein at each epoch we unfroze more layers and allowed gradient updates. We hoped for greater learning stability and better retention of general word embeddings from BERT's pretrained weights. After adjusting various hyperparamters (learning rate and dropout probability), our results were mixed at best. The results cited below are from training on the full model at learning rate 1e-5 over 10 epochs, where at each epoch we unfroze the top 10 most layers of BERT (ie. at the 10th iteration, 100/199 of the BERT layers were being updated) (Huo and Iwaihara, 2020).

Iteration 15 / Final: Noting that our hybrid Iteration 13 approach had thus far delivered the best results and also that there was considerable variation in performance across different training iterations, we reverted to the model specification for Iteration 13 and trained multiple times to select the highest performing iteration, which we utilized as our final dev leaderboard submission (Iteration 15) and test leaderboard submission (Final Test Leaderboard).

5.4 Results

	Sentiment Anal.	Paraphrase Det.	Sem. Text. Sim.	Overall
Baseline Model	**0.409**	**0.678**	**-0.09**	-
Iteration 1 (XAY 1)	0.409	**0.705**	**0.234**	**0.577**
Iteration 2 (XAY 2)	**0.442**	**0.710**	**0.345**	**0.608**
Iteration 3	0.356	-	-	-
Iteration 4	0.378	-	-	-
Iteration 5 (XAY 3)	0.438	**0.726**	**0.435**	**0.627**
Iteration 6	-	-	0.420	-
Iteration 7	-	-	0.415	-
Iteration 8 (XAY 4)	**0.446**	**0.834**	**0.614**	**0.696**
Iteration 9	0.435	0.811	0.587	-
Iteration 10 (XAY 5)	**0.494**	0.821	**0.717**	**0.725**
Iteration 11	-	-	0.654	-
Iteration 12	**0.527**	0.808	0.635	-
Iteration 13 (XAY 6)	0.525	0.821	0.717	**0.736**
Iteration 14	0.515	0.709	0.643	-
Iteration 15 (XAY 7)	**0.544**	0.827	0.716	**0.743**
Final Test Leaderboard	0.517	0.828	0.681	0.728

Table 1: Accuracy / Correlation Scores. Entries with a '-' indicate that task-specific model was not trained and evaluated at that iteration. Entries in bold indicate improvements in performance beyond any previous iterations of task. Iterations notated '(XAY *n*)' refer to iterations which were submitted to the dev leaderboard.

Broadly speaking, these results are in-line with our initial expectations. As a random baseline, accuracy scores for sentiment analysis would be roughly 0.2 (assuming equal likelihood of classification in any of the 5 classes), paraphrase detection scores would be roughly 0.5 (assuming binary classification with equal probability), and semantic textual similarity Pearson correlation would be roughly 0 (ie. no power to predict / identify any relationship between paired text inputs). Our results significantly outperform these random baselines and demonstrate that our model has predictive power on these tasks. Relative to our peers, our model also performs adequately, as reflected by our respectable standings on the leaderboards of both dev and test sets. We believe that our iterative approach and task-specific model architecture allowed us to make optimizations for each individual task without affecting performance on any other, ultimately culminating in desirable results for each of the task-specific models we iterated and implemented.

6 Analysis

We will divide our analysis into observations relevant to each of the 3 tasks, as our experimentation demonstrates that different approaches are optimal for each of them.

Sentiment Analysis: To begin with, we were disappointed by the underwhelming performance of the clustering techniques we implemented: k-means and spectral clustering. We suspect in hindsight, that implementing clustering within each batch may have been problematic. Given the batch size of 8, there is no guarantee that every batch will have at least one instance of each class in it. For batches where this is not the case, by imposing $k = 5$ classes in our clustering approaches, we may have been lowering the quality of embeddings as a result of incorporating misclassifications into them. Beyond this point, semantic analysis was the only task in which (1) the best results were obtained using a simple sequence of linear layer followed by non-linear activation (softmax in this instance), and (2) full model training. We believe that this is because semantic analysis, as opposed to the other two tasks, is based upon understanding the nuance associated with a single input, as opposed to measuring the contrast between paired inputs. BERT's existing layers and general architecture were designed with this task in mind (general langauge ability) and trained on next word prediction, a task which is well suited to capturing the nuance of inputs (Devlin et al, 2018). As a result, when finetuning for a similar task, in this case sentiment analysis, the embeddings generated from BERT finetuned on the full model and put through a simple linear layer and non-linear activation were optimal.

Paraphrase Detection: The primary method we employed for paraphrase detection was the inclusion of a sequential bidirectional LSTM followed by a linear layer. Specifically, we took the BERT embeddings for the two inputs in the input pair and transformed them through this bidirectional LSTM before taking the absolute difference of the resulting embeddings and using them as input to the final linear layer. The interesting observations here are the choice of neural network and the choice of embedding combination that led to the best result. In regards to the former, while the STS task performed best with a 1D Convolutional Layer, paraphrase detection performed best with a bidirectional LSTM. While both of these architectures help capture and learn non-linearities in the embeddings beyond that which a linear layer might be able to, LSTM may have worked better in this context because it is particularly well suited for sequential tasks, such as paraphrase detection, where the sequence of words in either input matters more (Hochreiter and Schmidhuber, 1997). As for the latter, we found that taking absolute difference was optimal when compared to alternatives such as concatenation of the embeddings of pairwise multiplication; this is primarily because absolute difference highlights the contrast in embeddings more than concatenation while multiplication may lead to skewed representations for higher magnitude embedding entries, biasing the combined embedding (Mao et al, 2021).

Semantic Textual Similarity: For this task, we employed a variety of techniques to varying effect before settling on our final optimal one. Firstly, we attempted to augment training data quality and quantity. To this end, we obtained a second task-specific dataset, SemEval's SICK dataset, and appended it to the standard provided one for training. We found that although this roughly doubled the amount of training data, the model actually performed slightly poorer as a result. We theorize that it may be due to differences in how the datasets were scored, despite the fact they came from the same source, demonstrating the importance of training data quality (Lin et al, 2022). As it relates to improving data quality, we attempted standard preprocessing techniques such as removing all punctuation and converting all inputs to lowercase characters in the datasets (Fan et al, 2021). This did not improve performance either, however, since BERT is pre-trained on properly formatted

data and finetuning on a small quantity of stripped down data cannot overcome the learned word embeddings from pretrained BERT (Devlin et al, 2018). The second notable observation was that this task was best completed using a 1D convolutional layer followed by a linear layer, as opposed to simple sequential linear layers or an LSTM approach, as was the case for the other two tasks. This is likely due to CNN's adeptness at identifying spatial hierarchies within features that allows it to better codify the differences between embeddings for this task (Kim, 2014).

7 Conclusion

In conclusion, we discovered that each of the three tasks required different finetuning approaches. Specifically, the methods which we found most successful were: sequential linear layer followed by softmax for sentiment analysis, bidirectional LSTM on individual embeddings followed by absolute difference combined embeddings passed through a linear layer for paraphrase detection, and 1D convolutional network on individual embeddings followed by absolute difference combined embeddings passed through a linear layer for semantic textual similarity.

In addition to these findings, we were able to perform well on both leaderboards while also exploring a wide variety of other finetuning techniques and gaining a greater appreciation of the specific contexts in which techniques are useful alongside contexts in which their limitations prevent them from improving performance.

At the same time, we recognize that there are significant limitations to our work. To begin with, using BERT as a baseline model means that any observations we made regarding technique efficacy can only be said in the context of the pretrained BERT model specifically; using a different pretrained architecture and its associated embeddings may lead to different results for different methodologies. Secondly, we defined task performance very narrowly, only benchmarking it against one dataset per task. As such, the performance of our methods and techniques cannot be generalized to the tasks of sentiment analysis, paraphrase detection, and semantic textual similarity broadly, but must be understood in the context of our narrow evaluation methodology.

Naturally, these limitations present avenues for future work. It would be meaningful to analyze how well our implemented techniques perform when a different baseline model is selected instead of BERT, such as GPT, (Radford et al, 2018), XLNet (Yang et al, 2019), T5 (Raffel et al, 2020), etc. Equally interesting would be to evaluate how the techniques performed when benchmarked against different datasets. Analyzing the similarities and/or differences in performance across these shifting variables will allow for more powerful conclusions regarding technique efficacy in broader contexts.

8 Ethics Statement

Natural language processing in general is rife with ethical dilemmas that bring into focus the dangers posed by further work in the area and highlight the necessity for an awareness of, and concern for, ethics in the space. For our project in particular, training models with the goal of adeptness at sentiment analysis, paraphrase detection, and semantic textual similarity prediction may lead to:
Inherited Biases: Models such as the one we finetuned are trained on existing data. When data is biased, skewed, or otherwise miscolored, the models trained on it will inherit these biases, which may either exacerbate existing biases in society or introduce new ones. To mitigate this, it is important to analyze and filter the training data which we utilize to remove any potential biases before they are learned by the model (Mehrabi et al, 2022).
Misuse: Training a model which has near human-level performance at these tasks generally presents as many opportunities for misuse as it does for benevolent ones. In the hands of bad actors, such models could be used to generate deepfakes, carry out scams, impersonate others, etc. To mitigate this, it is important to limit the model (ie. train it in such a way that it rejects malevolent requests) and/or potentially limit access depending on its capabilities (Brundage et al, 2018).
Lack of Consent: Often, training data for these models are obtained through web-scraping practices that take individual data without their expressed knowledge or consent. This is potentially a violation of inidvidual privacy and should be mitigated through regulation and/or scrutiny of data sources and complication methodologies prior to use in training (Shokri et Shmatikov, 2015).
This list is non-exhaustive and there are many other ethical dilemmas which should be taken into account when developing natural language processing models.

References

1. Benesty, J., Chen, J., & Huang, Y. (2008). On the Importance of the Pearson Correlation Coefficient in Noise Reduction. IEEE Transactions on Audio, Speech, and Language Processing, 16(4), 757-765. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4459449
2. Brundage, M., Avin, S., Clark, J., Toner, H., Eckersley, P., Garfinkel, B., Dafoe, A., Scharre, P., Zeitzoff, T., Filar, B., Anderson, H., Roff, H., Allen, G. C., Steinhardt, J., Flynn, C., Ó hÉigeartaigh, S., Beard, S., Belfield, H., Farquhar, S., Lyle, C., Crootof, R., Evans, O., Page, M., Bryson, J., Yampolskiy, R., & Amodei, D. (2018). The malicious use of artificial intelligence: Forecasting, prevention, and mitigation. arXiv preprint arXiv:1802.07228. https://arxiv.org/pdf/1802.07228
3. Cer, D., Diab, M., Agirre, E., Lopez-Gazpio, I., & Specia, L. (2017). SemEval-2017 Task 1: Semantic Textual Similarity-Multilingual and Cross-lingual Focused Evaluation. arXiv preprint arXiv:1708.00055. https://arxiv.org/abs/1708.00055
4. Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805. https://ar5iv.labs.arxiv.org/html/1810.04805
5. Ding, N., Qin, Y., Zhang, Z., Liu, X., Zhao, W. X., & Zheng, H. T. (2023). Parameter-efficient fine-tuning of large-scale pre-trained language models. Nature Machine Intelligence. https://www.researchgate.net/publication/368944790_Parameter-efficient_fine-tuning_of_large-scale_pre-trained_language_models
6. Dolan, W. B., & Brockett, C. (2005). Automatically Constructing a Corpus of Sentential Paraphrases. Proceedings of the Third International Workshop on Paraphrasing (IWP2005). https://aclanthology.org/I05-5002.pdf
7. Fan, C., Chen, M., Wang, X., Wang, J., & Huang, B. (2021). A Review on Data Preprocessing Techniques Toward Efficient and Reliable Knowledge Discovery From Building Operational Data. Frontiers in Energy Research, 9, 652801. https://www.frontiersin.org/articles/10.3389/fenrg.2021.652801/full
8. Han, X., Zhang, Z., Ding, N., Gu, Y., Liu, X., Huo, Y., Qiu, J., Zhang, L., Han, W., Huang, M., Jin, Q., Lan, Y., Liu, Y., Lu, Z., Qiu, X., Song, R., Tang, J., Wen, J., Yuan, J., Zhao, W. X., & Zhu, J. (2021). Pre-Trained Models: Past, Present and Future. arXiv preprint arXiv:2106.07139. https://ar5iv.labs.arxiv.org/html/2106.07139
9. Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. Neural Computation, 9(8), 1735-1780. https://www.bioinf.jku.at/publications/older/2604.pdf
10. Huo, H., & Iwaihara, M. (2020). Utilizing BERT pretrained models with various fine-tune methods in subjectivity tasks. Proceedings of the 12th DEIM Forum. https://db-event.jpn.org/deim2020/post/proceedings/papers/G1-1.pdf
11. Janocha, K., & Czarnecki, W. M. (2017). On Loss Functions for Deep Neural Networks in Classification. arXiv preprint arXiv:1702.05659. https://arxiv.org/pdf/1702.05659
12. Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. arXiv preprint arXiv:1408.5882. https://arxiv.org/abs/1408.5882
13. Krishna, K., & Murty, M. N. (1999). Genetic K-means algorithm. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 29(3), 433-439. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=764879
14. Lin, J., Zhang, A., Lécuyer, M., Li, J., Panda, A., & Sen, S. (2022). Measuring the Effect of Training Data on Deep Learning Predictions via Randomized Experiments. Proceedings of the 39th International Conference on Machine Learning, PMLR 162:13468-13504. https://proceedings.mlr.press/v162/lin22h.html
15. Luo, C., Zhan, J., Wang, L., & Yang, Q. (2017). Cosine normalization: Using cosine similarity instead of dot product in neural networks. arXiv preprint arXiv:1702.05870. https://arxiv.org/pdf/1702.05870
16. Mao, A., Mohri, M., & Zhong, Y. (2023). Cross-Entropy Loss Functions: Theoretical Analysis and Applications. Proceedings of the 40th International Conference on Machine Learning, in Proceedings of Machine Learning Research, 202:23803-23828. Available from https://proceedings.mlr.press/v202/mao23b/mao23b.pdf

17. Mao, R., Lin, C., & Guerin, F. (2021). Combining Pre-trained Word Embeddings and Linguistic Features for Sequential Metaphor Identification. arXiv preprint arXiv:2104.03285v1. https://arxiv.org/pdf/2104.03285v1

18. Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., & Galstyan, A. (2022). A survey on bias and fairness in machine learning. arXiv preprint arXiv:1908.09635. https://arxiv.org/pdf/1908.09635

19. Ng, A., Jordan, M., & Weiss, Y. (2001). On Spectral Clustering: Analysis and an Algorithm. Advances in Neural Information Processing Systems, 14, 849-856. https://proceedings.neurips.cc/paper_files/paper/2001/file/801272ee79cfde7fa5960571fee36b9b-Paper.pdf:

20. Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving language understanding by generative pre-training. OpenAI. https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf

21. Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., ... & Liu, P. J. (2020). Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. Journal of Machine Learning Research, 21(140), 1-67. https://arxiv.org/abs/1910.10683

22. Ren, Y., Pu, J., Yang, Z., Xu, J., Li, G., Pu, X., Yu, P. S., & He, L. (2022). Deep Clustering: A Comprehensive Survey. arXiv preprint arXiv:2210.04142. https://arxiv.org/pdf/2210.04142

23. Shokri, R., & Shmatikov, V. (2015). Privacy-preserving machine learning: Threats and solutions. Proceedings of the 2015 ACM SIGSAC Conference on Computer and Communications Security (CCS '15), 19-30. https://doi.org/10.1145/2810103.2813687

24. Szandała, T. (2020). Review and Comparison of Commonly Used Activation Functions for Deep Neural Networks. arXiv preprint arXiv:2010.09458. https://arxiv.org/pdf/2010.09458

25. Udell, M., Horn, C., Zadeh, R., & Boyd, S. (2015). Generalized Low Rank Models. Manuscript in preparation. Stanford University. https://arxiv.org/pdf/1410.0342

26. Walaa Medhat et al. (2014, May 27). Sentiment analysis algorithms and applications: A survey. Ain Shams Engineering Journal. https://www.sciencedirect.com/science/article/pii/S2090447914000550

27. Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., & Le, Q. V. (2019). XLNet: Generalized Autoregressive Pretraining for Language Understanding. arXiv preprint arXiv:1906.08237. https://arxiv.org/abs/1906.08237

28. Yin, W., Kann, K., Yu, M., & Schütze, H. (2017). Comparative study of CNN and RNN for natural language processing. arXiv preprint arXiv:1702.01923. https://arxiv.org/pdf/1702.01923