miniBERT and Multitasking: An Architectural Analysis

Stanford CS224N Default Project

Jack Michaels
Department of Computer Science
Stanford University
jackfm@stanford.edu

Mentor: Gabriel Poesia Reis e Silva

Abstract

BERT, the seminal transformer architecture outlined in Devlin et al. (2018), has galvanized the progress of recent language models, paving the way for generational technologies such as the recent advancement of Generative Pre-trained Transformers (GPT) models (OpenAI, 2023). This paper investigates the performance of BERT's architecture, implementing a simplified version of BERT referenced canonically as 'miniBERT'. Leveraging miniBERT, three sentence embedding tasks will be given to our model to assess multitasking performance and effective architectures. Our findings suggest that miniBERT embeddings capture significant underlying understanding of utterances, though they are limited due to their lack of cultural and context dependent comprehension. Then, through an architectural exploration this paper suggests that the architectures which effectively capitalize on BERT embeddings need not be unmanageably complex. Finally, we outline promising future explorations to improve model accuracy and give insight into future potential architectures to contrast the associated difficulties of our multitasking performance.

1 Introduction

Multitasking is the practice of training for two or more tasks within a single deep learning model, often separated by pre-training and fine-tuning phases (Ruder, 2017). Classical models have often approached tasks in a more direct way, hyper-optimizing one model to successfully achieve satisfactory performance results on only one particular metric. While these approaches are viable, multitasking models offer nuance by pre-training a shared model between all supplied tasks to garner a more general view on the discipline at hand, be it natural language processing (Collobert and Weston, 2008), speech recognition (Deng et al., 2013), or more.
From here, fine-tuning model layers on specific tasks can be done to steer the general model into acceptable performance on the supplied tasks. This project employs multitasking methodologies by utilizing the miniBERT implementation as a pre-trained general model to generate sentence embeddings for input into three domain related tasks: sentiment analysis, paraphrase detection, and semantic textual similarity. This paper then goes to explore potential model architectures for use in these various tasks, suggesting which architectures viably utilize the information captured in miniBERT generated embeddings.

1.1 Sentiment Analysis

Sentiment analysis is the task of predicting labels from 0 to 4 the polarity of a given utterance. Such a task has wide reaching commercial applications; determining instantaneously how customers are responding to products as a main one. This task is highly non-trivial, the model needs to analyze not

just word embeddings but their interplay amongst each other and significance in the broader context as a whole. For instance, given the 5 star review "I haven't laughed that hard in years!", the model needs to first look past the shallow significance of phrases like "I haven't laughed" and "in years" to secondly recall that laughing is good, and to finally make the leap to realize that not having laughed for years and finally laughing implies the existence of a positive experience, thus making the review positive.

1.2 Paraphrase Detection

Paraphrase detection is the task of predicting whether a given utterance is a paraphrase of some other given utterance. Applications exist in academic settings to thwart plagiarism, or in commercial question answering sites such as Quora where repeat questions must be organized and combined for clarity. This task tests the ability of our pre-trained model to capture effective sentence embeddings since any two utterances meaning nearly the same thing should have a correspondingly similar embedding.

1.3 Semantic Textual Similarity

Finally, semantic textual similarity is the task of predicting, on a 0-5 scale, the semantic similarity between two utterances. As this task is highly related to paraphrase detection, it also has a similar subset of applications with a newly granted higher granularity. Difficulty arises due to the granularity provided by the 0-5 scale, meaning the model must not only understand the underlying syntactic similarity between utterances but grasp all facets of an underlying utterance to accurately assess the degree of similarity.

2 Related Work

Architectures vary wildly depending on the task at hand and the available resources of specific articles. Due to extensive documentation of research done in the area, this project in particular heavily leveraged various implementations of sentiment analysis when determining a valid implementation strategy. Papers outline implementations of sentiment analysis via sentence type classification involving BiLSTM-CRF and CNN models (Chen et al., 2017), reflecting the implementations investigated in this paper by leveraging RNN models through miniBERT transformers and subsequently fine-tuning on CNN models. Ensemble models were explored in Alsayat (2021) to substantial efficacy, outperforming any individual model. From there, sentiment analysis based literature became increasingly interested in embedding architectures. A paper investigating sentiment-enhanced word embeddings powered through various RNN architectures (TextRNN, TextRCNN, TextBiRCNN, and more) proved effective but ultimately underwhelming in relation to this paper due to resource restrictions faced when attempting to implement the data heavy word vector based approaches (Li et al., 2022).
Due to paraphrase detection's immediately obvious similarity between semantic simlarity, the literature review primarily focused on paraphrase detection. Work has been conducted on implementing a BERT forward approach with CV-XGBoost, CV-CatBoost, TF-IDF-XGBoost, and TF-IDF-CatBoost methods to supplement the base BERT model achieving 97 percent accuracy (Chandra and Stefanus, 2020). This model attempted to learn on BERT and LSTM embedding representations supported by the various boost algorithms presented in the paper. Ultimately, the findings suggested that BERT implementations were nearly 20% more effective, supporting the implementation strategy of this paper. The comprehensive literature review by Zhou et al. (2022) on paraphrase identification presented this paper with foundational and inspirational architectural design insights to be incorporated in the later portion of this project, often directly describing the impacts particular infrastructural choices have on the model performance like RNN implementations. Finally, less complex models utilizing dependency parsing to generate effective embeddings have been utilized to great effect, providing efficient model architectures to consider (Chi et al., 2021).

3 Approach

The goal of this paper is to assess the performance of unique architecture styles on multitasking off BERT embeddings. Baseline architectures remained rather trivial while downstream task specific architectures incorporated more sophisticated model architectures. As a quick introduction without

much of an algorithmic explanation, as mentioned above miniBERT drew heavy inspiration from the seminal BERT model. miniBERT incorporated BERT transformer layers consisting of multi-headed attention, additive and normalization layers, and feed-forward layers. These bidirectional transformer layers formed the encoder for our miniBERT architecture which itself acted as the pre-trained portion of our multitasking model. For all of the following architectures, cross entropy loss was used for simplicity.

3.1 Baseline

The baseline architecture was chosen to assess the inherent capabilities of miniBERT with as minimal complex fine-tuning possible. As such, a simple architecture was chosen of only respective linear layers implemented for each task. To provide some form of general pre-training the outputted BERT embeddings, a linear layer with 307,200 parameters was used to funnel the 768 feature miniBERT embedding output into a more useful and efficient reduced 400 feature embedding space. Four hundred was chosen to reduce the embedding space by a factor of nearly 2. From there, all three tasks followed a similar architectural format with subtle differences for compatibility. The format involved either directly reduced the 400 feature embedding to its respective class number feature size and running softmax on that to generate probabilities, or reducing the 400 feature embedding to an even smaller embedding space and running cosine similarity on those embeddings. This smaller embedding space had 13 features to allow the model to place similar embeddings near one another more easily.

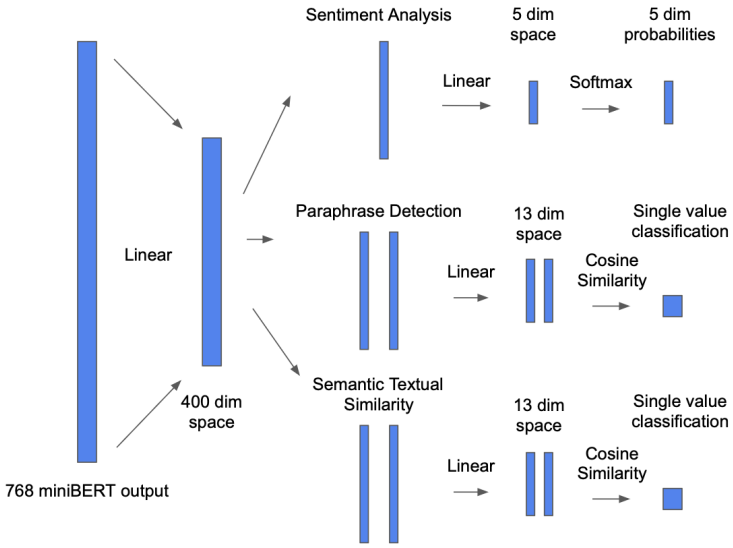

Figure 1: Baseline model architecture

3.2 Experimental Architectures

3.2.1 Recurrent Based Architecture

Due to the context dependent tasks, recurrent based architectures seem to offer a promising avenue of exploration as entertained in the literature (Chen et al., 2017). This architecture exploits recurrent based methodologies for semantic textual similarity in an attempt to examine if the model had any supplemental exploitation of miniBERT embedding information. To upscale the miniBERT embedding output to prepare for recurrent network input, the model begins with a pre-train convolutional upscaling using a filter size of 3 in an attempt to capture low level patterns in the embedding input and adding another dimension effectively transforming the singular embedding into 8 embeddings. ReLU was capitalized as an efficient non-linear activation function. This inputted into the specific semantic textual similarity model, which utilized bidirectional LSTM cells for encoding the inputted embeddings into a more useful lower 32 feature embedding space. The final hidden state of the LSTM was used as its output. From there, the output was passed into a linear layer to funnel values into a single classification. The goal of this architecture was for the model to embed general sentence

significance into the 8 embeddings, learn the interplay between these embeddings through the LSTM, and then output a corresponding predicting upon effective analysis.

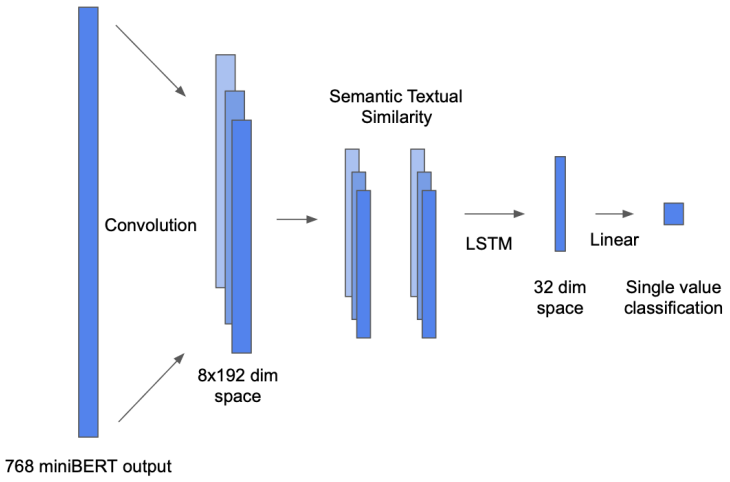

Figure 2: Sentiment Analysis Recurrent Architecture

3.2.2 Convolutional Based Architecture

Convolutional neural networks could theoretically ascertain patterns in embedding spaces to extrapolate important learning considerations. Another viable model thus involves heavy usage of convolutional neural networks, the final model we tested and implemented. As was done in the recurrent based architecture, a convolutional upscaling was performed initially to pretrain some model pattern recognition across the three tasks and to increase the amount of data prevalent. Upon completion, both sentiment analysis and semantic textual similarity ran upon convolutional architectures while paraphrase detection just employed a linear model due to the simplicity of its predictive range. For sentiment analysis, an additional convolutional upscaling was performed with a pool layer in an attempt to allow the model to train upon as much **important** data as possible. A similar upscaling was performed in the semantic textual similarity model. Afterwards, linear layers were used to downscale the data into an effective embedding space, upon which softmax was run on sentiment analysis to output probabilities and cosine similarity was run on semantic textual similarity for scale predictions. To allow for the generalization required of sentiment analysis mentioned in the introduction batch normalization and dropout were used as regularization.

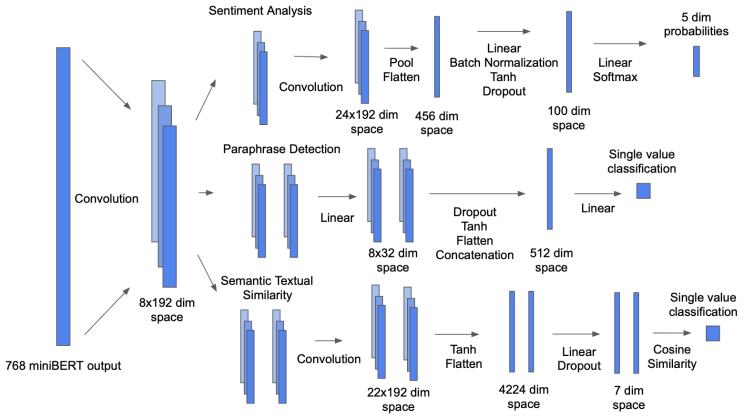

Figure 3: Convolutional Based Architecture

4 Experiments

4.1 Datasets

Three datasets were used for the corresponding three tasks: the Stanford Sentiment Treebank for sentiment analysis, the Quora dataset for paraphrase detection, and the SemEval STS Benchmark dataset for semantic textual similarity prediction.

4.1.1 The Stanford Sentiment Treebank Dataset

The Stanford Sentiment Treebank consists of 11,855 sentences from movie reviews. Each review contains a label from 0-4 describing the sentiment of the review with 0 being the most negative review possible and 4 being the most positive review possible.

"A bit too derivative to stand on its own as the psychological thriller it purports to be." (2)
"Staggeringly dreadful romance." (0)
Example 1: Sentences From the Stanford Sentiment Treebank Dataset and Their Ratings

This paper utilizes 8,544 examples in the training set, 1,101 examples in the development set, and 2,210 examples in the test set.

4.1.2 The Quora Dataset

The Quora dataset consists of 400,000 pairs of questions with associated binary labels indicating whether they are paraphrases of one another or not.

"Will the pound sterling keep weakening?"
-and-
"Will the english pound keep falling?"
Example 2: Valid Paraphrases From the Quora Dataset

Due to the extreme size of the dataset and computational resource restrictions, only a representative subset of the dataset was trained on. This paper utilizes 8,544 examples in the training set, 4,544 examples in the development set, and all 40,431 examples in the test set. 37% of the training subset are valid paraphrase sentence pairs with the remaining being invalid paraphrase sentence pairs demonstrating the subset is a representative subset.

4.1.3 The SemEval STS Benchmark Dataset

The SemEval STS Benchmark dataset consists of 8,628 different sentence pairs labelled from 0-5 with a rating of 0 representing completely unrelated sentence pairs and a rating of 5 meaning equivalent sentences.

"A cat plays in his water."
-and-
"The cat fished a straw out of his water dish."
Example 3: Sentence Pairs From the SemEval STS Benchmark Dataset With a 3.5 Rating

This paper utilizes 6,041 examples in the training set, 864 examples in the development set, and 1,726 examples in the test set.

4.2 Evaluation method

Following Chen et al. (2017), accuracy was chosen as the appropriate evaluation metric for sentiment analysis and paraphrase detection due them being classification type tasks. Since the sentiment textual similarity task labels on a scale of 0-5, accuracy is an invalid metric to assess performance. Pearson product-moment correlation coefficients was used instead following Atoum (2019) to correctly assess a model that consistently scores within a range of the true label instead of rewarding a model that randomly happens to score closely to the true label infrequently.

4.3 Experimental details

For multitasking, models are trained in two phases: pre-training and fine-tuning. Pre-training was conducted with a batch size of 64 and learning rate of 0.001 for quicker model convergence. Training was done over ten epochs with AdamW optimization and was conducting through Google's premium compute resources on an A100 NVIDIA GPU. Even with the diminished dataset size, pre-training for the baseline, recurrent model and convolutional model had training time estimates around 1 hour 39 minutes, 2 hours 20 minutes, and 2 hours 34 minutes respectively.
Fine-tuning was conducted with a batch size of 8 and learning rate of 10^{-6} to allow for longer and more delicate training reflecting the important of the fine-tuning phase on accuracy. To allow for sufficient model convergence, training was done over 25 epochs again with AdamW optimization and again trained through Google's premium compute resources. Due to the lower batch size and higher epoch number, training for the baseline, recurrent model and convolutional model had training time estimates around 2 hours 37 minutes, 3 hours 15 minutes, and 3 hours 33 minutes respectively for the fine-tuning phase demonstrating the necessity for higher compute resources.
Note that a pre-training learning rate of 0.1 was attempted to determine if training could be done more efficiently when the model is largely uninitialized. However, such a learning rate caused no convergence though additional hyperparameter search possibilities are outline in the Conclusion / Future Work section of this paper. For both of these architectures, a dropout rate of 0.3 was used for effective regularization.

4.4 Results

4.4.1 Development Training

Development training details the improvement over time and potential strength of the various potential architectures. Figure 4 illuminates the capabilities of the convolutional based architecture compared to the recurrent based and baseline architectures. Analyzing the performance during pre-training on paraphrase accuracy and sentiment analysis/semantic textual similarity, both the recurrent and convolutional based architectural approaches imply effective post miniBERT embedding processing through the convolutional layer described in the architecture section above. Pre-training paraphrase detection was underwhelming to say the least, and this was likely caused by the pre-training process learning to predict all negative or all positive values instead of learning effective predictive techniques. This is a clear area for improvement, showcasing the effectiveness of pre-training a specific miniBERT embedding model instead of a general pre-trained miniBERT implementation.
During fine-tuning, Figure 4 illuminates the importance of computational resources. All three architectures share a slowly increasing sentiment classification accuracy instead of flat lining accuracy (like semantic textual similarity), demonstrating the potential for increased accuracy through increased training time and computational resources.

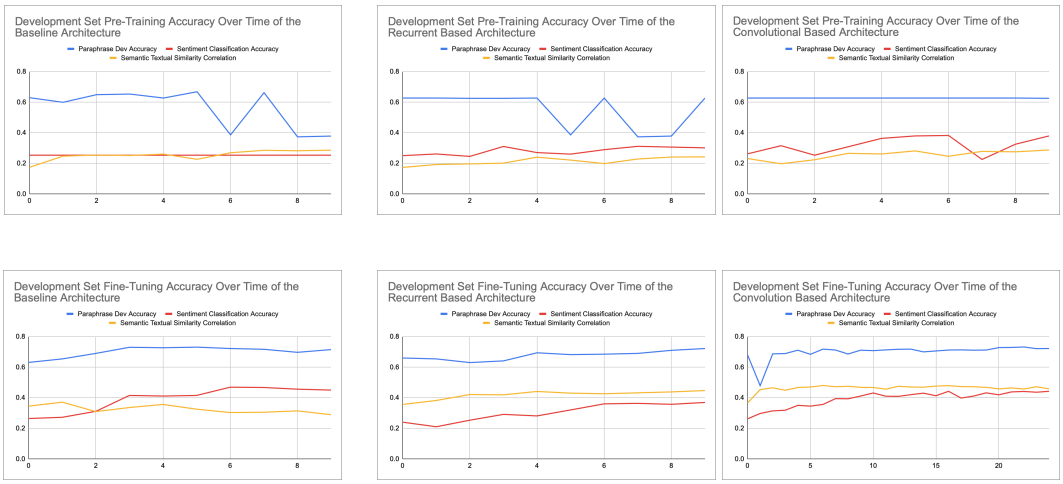

Figure 4: Training Graphs for the Baseline, Recurrent Based, and Convolution Based Architectures Ordered Vertically by Pre-Training and Fine-Tuning.

4.4.2 Test Results

Though recurrent based architectural approaches were theorized to potentially exceed convolutional based approaches due to their ability to decipher context and far separated sentence dependencies, their unwieldy complexity compared to the computational resources and subsequently small datasets suggests that convolutional based approaches offer more effective means. Convolutional based architectures pose nearly at 50% increase in semantic textual similarity correlation and also slightly increase the effectiveness of paraphase detection. This is voided by the fact that the baseline performed marginally more accurately than both the recurrent and convolutional based architectures on sentiment analysis. A discussion of the reasoning behind this is posed in the Analysis section of this paper. Nonetheless, this data suggests that baseline performance unexpectedly lives up to both the recurrent and convolutional based approaches, though convolutional based approaches offer nuance complexity which ultimately exceeds the capabilities of the baseline overall.

Table 1: Test Accuracy for the Three Architectures

	Baseline	Recurrent	Convolutional
Paraphrase Accuracy	0.727	0.722	**0.735**
Semantic Textual Similarity Correlation	0.304	0.341	**0.457**
Sentiment Accuracy	**0.469**	0.377	0.381

Table 2: Final Train Accuracy for the Three Architectures

	Baseline	Recurrent	Convolutional
Paraphrase Accuracy	0.985	0.854	0.928
Semantic Textual Similarity Correlation	0.795	0.576	0.803
Sentiment Accuracy	0.614	0.659	0.639

5 Analysis

Table 1 demonstrates the effectiveness of convolutional based architectures in paraphrase accuracy and semantic textual similarity correlation. However, sentiment analysis was dominated by the baseline model. Such domination likely doesn't come from the baseline performing particularly well, but from the recurrent and convolutional based architectures overfitting to the training data as seen in Table 2 and subsequently performing poorly on sentiment classification. Though regularization was attempted in both the recurrent and convolutional based models, the lacking computational resources affecting the size possibility of the data is likely the culprit. The Conclusion / Future Work section details possible extensions to resolve this overfitting issue. Furthermore, none of the models were able to reach a significantly high performance on the training sets, demonstrating higher computational resources are required to fully exploit the nuance of the provided dataset, yet this paper was bounded by lacking computational power.

In regards to the standout performance of paraphrase detection when compared to semantic textual similarity correlation across the board, two factors are to be considered. Firstly, though paraphrase detection and semantic textual similarity are similar in regards to their both learning on the underlying similarity of given utterances, the scale associated with semantic textual similarity correlation poses significant learning challenges since paraphrase detection can incorrectly get labels correct with much higher chance, thus exploding the accuracy falsely. Furthermore, nearly all models experienced implementation challenges when forcing an output to be between 0-5. Though cosine similarity was used and then exploded to be between 0-5, it is much harder for the model to achieve a label of 5 or 0 than a label of 2.5 since the model must align all 13 features or unalign them to achieve such a polarizing score. Thus, the accuracy of around 0.45 likely suggests that only labels from 1-4 were possible. Implementing some equalizing labeling technique will likely increase the accuracy drastically by allowing the model to label 0-5 in a less cumbersome form.

Finally, with regards to the low accuracy of the sentiment analysis task across the board, we attribute that simply to the complexity of the task at hand. Predicting sentiment is a highly context dependent setting, nuanced by sarcasm and cultural slang which drastically influences one's underlying understanding of any given utterance. Though recurrent and convolutional models can theoretically capture nuance and context, they can not do so outside of their training boundaries and therefore cannot accuracy capture new cultural slang terminology or even realize the overarching theme of analyzing on specifically movie reviews as the reader may pick up on. This demonstrates the importance of pre-training and creating a model which can effectively understand the nuance sentences in the broader context of the world and thus understand sarcasm and slang terminology, though due to time restrictions such improvements were considered out of scope for this paper.

6 Conclusion / Future Work

This paper suggests that convolutional based neural networks are the most effective at deciphering the complex sentence embeddings outputted from BERT based embedding models, though more computational resources, tasks, and architecture exploration is required to solidify that precedent. An exploration of utilizing word embeddings as opposed to sentence embeddings may pose as possible future research avenues due to their higher nuance than sentence embeddings, which may have contributed to the convolutional based neural networks dominance over recurrent based models.
In addition to these future research avenues, potential model improvements for the future reassessment of these figures are numerous. Potential for an immediate hyperparameter search exists in the exploration of the filter sizes for the convolutional layers, the linear layer output sizes, and the type of recurrent neural network used. Yet, potential exists primarily in expanding the computational resources and subsequently the training time allowed for the various architectures. This allows not only for the utilization and incorporation of all of the training data into the model but for the allowance of utilizing all of the dataset and additional datasets (for instance expanding to all 400,000 sentence pairs of the Quora dataset). Furthermore, datasets can be combined through new and more interesting loss functions, for instance multiple negatives loss by taking example sentences from say the Quora dataset and applying it to the semantic textual similarity task by simply applying a pre-processing layer.

References

Ahmed Alsayat. 2021. Improving sentiment analysis for social media applications using an ensemble deep learning language model. *Arab J Sci Eng*, page 2499–2511.

Issa Atoum. 2019. Scaled pearson's correlation coefficient for evaluating text similarity measures. *Modern Applied Science*, 13:26.

Andreas Chandra and Ruben Stefanus. 2020. Experiments on paraphrase identification using quora question pairs dataset.

Tao Chen, Ruifeng Xu, Yulan He, and Xuan Wang. 2017. Improving sentiment analysis via sentence type classification using bilstm-crf and cnn. *Expert Systems with Applications*, 72:221–230.

Xiaoqiang Chi, Yang Xiang, and Ruchao Shen. 2021. Paraphrase detection with dependency embedding. In *2020 4th International Conference on Computer Science and Artificial Intelligence*, CSAI 2020, page 213–218, New York, NY, USA. Association for Computing Machinery.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: deep neural networks with multitask learning. In *International Conference on Machine Learning*.

Li Deng, Geoffrey E. Hinton, and Brian Kingsbury. 2013. New types of deep neural network learning for speech recognition and related applications: an overview. *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8599–8603.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Qizhi Li, Xianyong Li, Yajun Du, Yongquan Fan, and Xiaoliang Chen. 2022. A new sentiment-enhanced word embedding method for sentiment analysis. *Applied Sciences*, 12(20).

OpenAI. 2023. Gpt-4 technical report.

Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks.

Chao Zhou, Cheng Qiu, and Daniel E. Acuna. 2022. Paraphrase identification with deep learning: A review of datasets and methods.