Exploring LSTM minBERT with DCT

Stanford CS224N Default Project
Mentor: Olivia Lee

Tina Wu
Department of Electrical Engineering
Stanford University
xinfangw@stanford.edu

Benita Wong
Department of Computer Science
Stanford University
benitaw@stanford.edu

Abstract

Our project aims to investigate various methods to jointly improve BERT model performance on three downstream tasks - SST, PARA, STS - in a manner that strikes a balance between exploiting established procedures (LSTM, Pair Encoding) and exploring novel methods (exploiting frequency information via Discrete Cosine Transform (DCT) related techniques, and usage of POS Tags). The enhanced model which exploits the aforementioned established procedures achieved the greatest improvement in overall dev score **(+0.12)** over a simple baseline, while the best performance on individual tasks (SST: 0.530, PARA: 0.866, STS: 0.840) came from enhanced models further augmented with DCT-related techniques. Addition of POS Tags was not found to have contributed to model performance. An analysis into the various methods attempted and their results are presented in this paper. Our best model attained test accuracies of **0.525** (SST), **0.861** (PARA) and Pearson's correlation score of **0.829** (STS), culminating in an overall score of **0.767**.

1 Introduction

Our project aims to investigate various methods to jointly improve BERT model performance on three downstream tasks - sentiment analysis (SST), paraphrase detection (PARA) and semantic textural similarity (STS) via transfer learning. To strike a balance between exploiting established procedures and exploring novel methods to improve model performance, our project first investigates the extent to which a combination of commonly adopted downstream task-specific model layers (LSTM) and procedures (sentence-pair encoding) help to improve model performance. This is followed by an investigation and analysis into whether the incorporation of additional frequency and language features in downstream models can lead to further improvements on the same three tasks.

The motivation behind this two-stage investigation stems from our recognition that, given the popularity of language models, there is already much exploration into ways to improve downstream model performance on language tasks. However, the proposed methods are often localised to a specific task - as they generally involve finetuning pre-trained language models on a singular supervised task - with less emphasis on whether these methods can be extended to other language tasks of interests or how they perform under multi-task learning. As such, there is intrinsic value in **first understanding how a combination of established procedures can jointly improve our BERT model performance on the three downstream tasks of interest via multi-task learning.** Beyond the inclusion of more established procedures, there also has been growing interest in figuring out if and how the use of frequency information and other language features in downstream models can lead to improvements in language tasks. This interest stems from the knowledge that language contains structure across different scales (e.g. word, sentence, paragraphs) (Tamkin et al., 2020) which can potentially be exploited via spectral filtering to assist in the performance of downstream tasks. Hence, our paper also seeks to perform a preliminary exploration and analysis into **whether the exploitation of frequency information and the addition of language features in our downstream models** can help with performance on our three tasks.

Through the various experiments conducted, our paper shows that an extended model which utilises LSTM and sentence pair encoding achieved test accuracies of 0.525 (SST), 0.861 (PARA) and Pearson's correlation score of 0.829 (STS). Our model also achieved an overall score of 0.767 on the dev set which is a big improvement (+0.12) over the simple baseline constructed using only linear layers. While our experiments with frequency features and filtering did not yield a higher overall dev score relative to the aforementioned extended model, our best performance(s) on individual tasks actually came from these frequency-augmented models. Contrary to our expectations, the addition of explicit incorporation of POS tags as features to downstream model did not appear to yield much benefit over the simple baseline. An analysis into the various experimented approaches and their corresponding performance will be presented in this paper.

2 Related Work

BERT was introduced by Devlin et al. (2018) and has since been used to obtain state-of-the-art performances on a multitude of natural language processing tasks. At its core, BERT is a language representation model specifically designed for the pre-training of deep bidirectional representations on unlabelled text with the intention being that the resultant pre-trained BERT model - which has learnt general semantic and syntactic information - can be fine-tuned with a single output layer to attain state-of-the-art models across a wide range of language tasks without the need for significant task specific modifications. Beyond fine-tuning BERT on a singular downstream task, it is possible to simultaneously fine-tune BERT on multiple downstream tasks by adding together the losses from tasks of interests (Bi et al., 2022). In view of this, our paper first starts by developing a simple baseline model (comprising only linear layers in each of the 3 downstream tasks) finetuned from pre-trained BERT model using the multi-task learning approach as a way to evaluate the limits of the pre-trained BERT model on our specific tasks. This method of utilising rich, unsupervised pre-training before finetuning on supervised tasks is more generally known as transfer learning and has been shown in machine learning research across multiple domains (computer vision, language understanding) to be a highly effectual approach at improving task specific performance Devlin et al. (2018).

2.1 Sentence Pair Encoding

BERT was pre-trained on the Next Sentence Prediction (NSP) task to help the model learn the relationship between two sentences. To do so, the sentence pair was first packed together in a single sequence - separated by a [SEP] token - and the input representation was then constructed from the packed sequence before being fed to BERT. This sentence-pair encoding approach to obtain an input representation of the pair has also been used in the fine-tuning on specific downstream language tasks such as Question-Answering Devlin et al. (2018), and can potentially be extended to other tasks wherein the relationship between a sentence pair is crucial such as paraphrase detection).

2.2 LSTM for Sentence Similarity

Due to the presence of memory cell units, LSTMs have been shown to be adept at learning long range dependencies. Mueller, J et.al proposed the usage of a Manhatten LSTM (MaLSTM) model that utilises what is - in essense - a singular LSTM (given tied weighs) to separately process each of the two sentence in the sentence pair. The hidden state extracted from the LSTM is then utilised as the vector representation for each sentence - from which similarity between the two representations can be computed. MaLSTM contains a very simple similarity function based in Manhatten distance to force the LSTM to capture semantic differences during the training procedure. MaLSTM was eventually able to achieve a high Pearson correlation score of 0.8822 on the SICK dataset. The simplicity (given the tied weights) and the high performance of this approach inspired the adoption of a similar model for our downstream STS task. (Mueller and Thyagarajan, 2016)

2.3 FNet and Language in a Prism

In exploring the intersection of advanced signal processing techniques and natural Language Processing, recent studies have showed the potential of leveraging Discrete Fourier Transforms (FT) and Discrete Cosine Transforms (DCT) to enhance language model's performance across various different tasks and improve the computational efficiency. The work presented in Tamkin et al. (2020) delve

into the structural hierarchy inherent in language and utilize spectral filters to generate hierarchical features. This approach filtered facilitates the disentanglement of information in existing embeddings to particular linguistic scale, such as word-level, sentence-level, or document-level, and trained models to learn more about particular scales. Concurrently, Lee-Thorp et al. (2022) presents the FNet which demonstrates that the Transformer encoder architectures can be sped up by substituting the self-attention sublayers with a simple, unparameterized Fourier Transform layer with limited accuracy costs. The FT sublayer applies one 1D Discrete Fourier Transform along the sequence dimension and one 1D Discrete Fourier Transform along the hidden dimension. This approach achieves 92-97% of the accuracy of BERT counterparts on the GLUE benchmark, but trains 80% faster on GPUs. These findings suggest that utilizing frequency features extracted by FT and DCT offers a path toward more efficient and scale-aware modeling architectures in NLP.

3 Approach

As per the project guidelines, we first implemented BERT with the Adam Optimizer and fine-tuned this pre-trained model on three downstream tasks using a multitask learning approach. To establish a benchmark against which improvements can be evaluated, we trained a simple baseline model consisting solely of linear layers. Following which, our approach is structured in a way that we methodologically answer **two questions** - targeted at improving model performance - via a series of extensions and their corresponding experimental results.

3.1 Extension Part 1: Enhancing the Baseline with Advanced Downstream Models

In a bid to improve model performance, we first sought to answer: **how much improvements can we yield over the simple baseline by implementing commonly adopted procedures to improve task specific performance?** These procedures include utilising a bi-directional LSTM layer in the downstream model (Fig. 1(a)) for the STS task to better capture the full context of the sentence (Pontes et al., 2018) as well as utilising sentence-pair encoding (Fig. 1(b)) for the PARA task. Particularly the use of sentence-pair encoding was inspired by the methodology utilised in (Devlin et al., 2018) to jointly pack Question-Answer pairs into a single sequence (separated by a special [SEP] token) using the BERT tokenizer. Given that the PARA task involves learning how two sentences are related to each other - which is something that is not directly captured by language modelling (Devlin et al., 2018), we believed that the PARA task would benefit from the use of sentence-pair encoding.

3.2 Extension Part 2: Incorporation of Frequency Information and POS Tagging

Upon establishing an improved model, our inquiry progresses to the second question: Can further enhancements be realized by **integrating frequency features** via techniques in signal processing, such as Discrete Fourier Transform (FT), see Equation (1), and Discrete Cosine Transform (DCT), see Equation (2), into the training process? These two transforms are applied directly to the embeddings generated by the minBERT model, transforming them from the time domain to the frequency domain. This methodological innovation aims to capture and leverage frequency-based features embedded within the linguistic data, an approach inspired by recent advances in signal processing applications in natural language processing, (Lee-Thorp et al., 2022) and (Tamkin et al., 2020).

$$X^{(k)} = \sum_{n=0}^{N-1} x^{(n)} e^{-i2\pi \frac{k}{N} n} \text{ for k = 0, 1, ...N-1} \tag{1}$$

$$f^{(k)} = \sum_{n=0}^{N-1} x^{(n)} cos\big[\frac{\pi}{N}(n + \frac{1}{2}k)\big] \text{ for k = 0, 1, ...N-1} \tag{2}$$

Our approach involves applying FT and DCT to the contextual embeddings derived from minBERT, immediately after obtaining the pooler output from the BERT model. The transformed embeddings undergo a frequency domain analysis where specific components, presumed to hold valuable linguistic patterns at varying scales, are isolated and potentially enhanced. This process hypothesizes that the frequency domain representation of text data can uncover latent structural features not readily obvious in the original embeddings.

Beyond frequency analysis, we are also interested in investigating if and how the explicit incorporation of linguistic clues such as POS tagging can help with the performance of the model on specific downstream task. Particularly, POS tagging has been shown to be useful for sentiment classification tasks (Kalaivani et al., 2018) Usop et al. (2017). However, if should be noted that most of the work done in this area do not utilise both BERT embeddings and POS tags as features for the downstream sentiment classification task concurrently. Hence, the effects of combining POS tags with BERT embedding on sentiment classification are still under-explored. There is - nonetheless - reason to believe that incorporating POS tags as a feature would help downstream model performance.

4 Experiments

4.1 Data

The four datasets provided for the default project—namely, the Stanford Sentiment Treebank, CFIMDB, Quora, and the SemEval STS Benchmark—are utilized both for implementations of the default project as well as for our project extensions.

4.2 Evaluation method

The sentiment analysis (SST) and paraphrase detection (PARA) tasks were assessed using accuracy, while the semantic textural similarity (STS) task was evaluated based on Pearson's correlation.

4.3 Experimental details

All models were generally fine-tuned using a learning rate of 1e-5 across 10 epochs, with a dropout rate of 0.3 and a weight decay of 0 unless otherwise specified. The loss functions selected for the SST, PARA and STS tasks were cross entropy, binary cross entropy and mean squared errors respectively. For the multitask training, three datasets were combined into a zipped dataloader, setting the batch sizes to 8 for SST, 32 for PARA and 8 for STS to account for the different dataset sizes and GPU RAM limitations. Furthermore, the losses from the 3 tasks, after adjusting for different batch sizes, were equally weighted in a $1:1:1$ ratio during multitask learning. Following are details of the experiments and the architecture of the model employed:

1. **Simple baseline:** The pooled output embeddings from BERT are inputted into three downstream models. Each model includes a single linear layer that processes the BERT embeddings to produce the required logits: 5 for the SST, and 1 for PARA and STS.
2. **Enhancing the Baseline with Advanced Downstream Models**: To augment the complexity and efficacy of the downstream task models—particularly for the PARA and STS tasks—two significant extensions were implemented.
    - **LSTM + Cosine Similarity for STS (Fig 1a):** The last hidden state embeddings from BERT for two sentences were input into a 2 layer bidirectional LSTM with hidden size of 300. The outputs are passed through a linear layer and subsequently mean pooled. The cosine similarity between these pooled outputs is then calculated. This approach is inspired by the work of Pontes et al. (2018); Mueller and Thyagarajan (2016).
    - **Pair Encoding for PARA(Fig 1b):** This approach involves jointly encoding the input sentences using the default BERT tokenizer. The sentences are concatenated into a single sequence, separated by a [SEP] token, and then inputted into the pre-trained BERT model to obtain a combined pooled output embedding. A tanh activation function is applied to this pooled output embedding before it is passed to a linear layer for further processing. This approach draws inspiration from the pair-encoding method utilised in the QA task outlined in Devlin et al. (2018).
3. **Frequency Embedding and Filtering (Fig 2):** To explore the potential of frequency embeddings, we designed the application strategy of one dimensional Discrete Fourier Transform [1] and type-II DCT [2] on minBERT's embeddings. The output from Fourier Transform has two parts, the real part and the imaginary part. In our implementation, we

[1]We use the `torch.fft.fft` function to do the transform.

[2]We use an external PyTorch library for computing the DCT `https://github.com/zh217/torch-dct`

only utilized the real part, following the implementation in Lee-Thorp et al. (2022). Since discarding the imaginary part of the results from Fourier Transform may discard the phase information of the input, we implemented the DCT as well, which focuses solely on real-valued cosine-based frequency components, providing an alternate lens on the frequency embedding content. Both two signal processing techniques, FT and DCT, were applied directly to the embeddings generated by the minBERT model. Moreover, we integrated a dynamic filter applied to the frequency embeddings, allowing the model to adaptively assign the weighting to each frequency component. Through this mechanism, we aim to refine the signal by emphasizing relevant features while attenuating those seemed extraneous, thereby enhancing the model's capacity to distill and exploit meaningful frequency signals from the data. For all the experiments in this extension focusing on frequency features, we set fine-tuned models using a learning rate of 1e-5 and weight decay of 1e-7 across 15 epochs.

4. **Incorporation of POS information (Fig 3):** To investigate whether the addition of POS information can help with the downstream SST task, spaCy [3] was utilised to convert the BERT tokenized words into their POS representation before this POS representation was converted into a one-hot representation. A learnable NNEmbedding layer was added as part of the downstream model for SST, which takes in this one-hot representation (spanning 18 dimensions) and outputs the 8-dim dense representation of the POS tags.

4.4 Results

The dev results of the various extensions incorporated are as shown in Table 1 (advanced downstream models), Table 2 (incorporation of frequency information) and Table 3 (incorporation of POS tags).

dev accuracy (%) / pearsons'	**Sentiment Analysis**	**Paraphrase Detection**	**Semantic Textual Similarity**	**Overall**
(1)Simple Baseline	0.514	0.728	0.377	0.6435
(2)Simple Baseline + LSTM for STS	0.52	0.761	0.824	0.731 *(+0.0875)*
(3)Simple Baseline + LSTM for STS + Sentence Pair Encoding for PARA	0.518	0.856	0.826	0.7623 *(+0.119)*

Table 1: Extension Part 1: Enhancing the baseline with advanced downstream models

dev accuracy (%) / pearsons'	**Sentiment Analysis**	**Paraphrase Detection**	**Semantic Textual Similarity**	**Overall**
(1)Simple Baseline	0.525	0.736	0.307	0.638
(2)Simple Baseline + Extension P1	0.517	0.863	0.840	0.767 *(+0.129)*
(3)Simple Baseline + DCT w filter	0.502	0.740	0.315	0.633 *(-0.005)*
(4)Simple Baseline + Extension P1 + DCT w filter for SST	0.514	0.858	0.837	0.763 *(+0.125)*
(5)Simple Baseline + Extension P1 + DCT w filter for PARA	0.521	0.854	0.840	0.766 *(+0.128)*
(6)Simple Baseline + Extension P1 + DCT w filter for STS	0.530	0.845	0.800	0.759 *(+0.121)*
(7)Simple Baseline + Extension P1 + DCT w filter for STS,STS,PARA	0.500	0.866	0.803	0.756 *(+0.118)*

Table 2: Incorporating Frequency Information (**15 epochs, weight decay = 1e-7**). *Note: Extension P1 refers to LSTM for STS and Pair Encoding for PARA*

dev accuracy (%) / pearsons'	**Sentiment Analysis**	**Paraphrase Detection**	**Semantic Textual Similarity**	**Overall**
(1)Simple Baseline	0.514	0.728	0.377	0.6435
(2)Simple Baseline + POS for SST	0.506	0.730	0.345	0.6316 *(-0.012)*

Table 3: Incorporation of POS Tags for SST task

Despite the DCT models (Table 2) displaying superior performance for individual tasks (SST Table 2 (6), STS Table 2 (5), PARA Table 2 (7)), their overall dev accuracy is slightly lower than the extended

[3] spaCy loaded with en_core_web_sm was utilised as the POS tagger `https://spacy.io/models/en`

model without DCT. Hence, the predictions from the simple baseline with LSTM and pair encoding extension (Table 2 (2)) on the test set were submitted to the leaderboard, wherein an **overall test score of 0.767 (STS: 0.525, PARA: 0.861, STS: 0.829)** was attained.

5 Analysis

5.1 LSTM and Sentence Pair Encoding

The relevant curves logged in wandb (Fig 4) highlight the differences between simple baseline, the simple baseline with LSTM and with sentence pair encoding during the training process. Intuitively, the addition of more complicated downstream models such as LSTM for STS (Fig 4 pink line) would result in lower train loss compared to the simple baseline (Fig 4 purple line) simply because there are now more parameters that the model can use to fit the training data. The corresponding increase in best dev accuracies seem to imply that this increase in complexity has not (yet) led to a serious over-fitting of the model to the train data. It is interesting to note that while both the purple and pink lines in the train_sts_corr graph both plateau at around 0.97, there is a huge difference in the performance of the respective models on the STS task wherein the simple baseline only achieved a pearson's correlation score of 0.377 on the dev STS set compared to the 0.824 that the addition of LSTM yielded. As mentioned in Section 2.2, this difference could potentially be attributed to the fact that LSTM's are able to learn long range dependencies which is likely crucial for tasks such as semantic textural similarity wherein one needs to look at the whole sentence before ascertaining the similarity in meaning. To illustrate this, select examples from the STS dev set and the predictions made using the different models are analysed.

Table 4 below contains two long sentence pairs as well as their corresponding ground truth values provided in the STS dev set and the logits predicted from the models with and without LSTM incorporated for the downstream STS task. Based on the ground truth values, it is observed that the second sentence pair (3.2) is more similar in meaning than that of the first sentence pair (2.8). From the logits output by the simple baseline model without LSTM, we see that it actually wrongly "predicts" that the second sentence pair is less similar than the first whereas the logits output by the model with LSTM correctly illustrates that the second sentence pair is more similar than the first one.

Sentence 1	**Sentence 2**	**Ground Truth**	**w/o LSTM**	**w LSTM**
Since early May, the city has received 1,400 reports of dead blue jays and crows, Jayroe said	Since early May, the city has received 1,400 reports, he said.	2.8	0.624	0.418
russian officials have called for a conference on the conventional forces in europe treaty to discuss ratification of the amended treaty.	antonov spoke the day before a conference on the conventional forces in europe treaty.	3.2	0.508	0.600

Table 4: Analysing Long Sentences STS

Conversely, for short sentence pairs (Table 5), we see that both models with and without LSTM do a decent job at predicting relative sentence similarities. It is interesting to note that, even for short sentences, the model comprising LSTM still does a better job at predicting the relative extents of similarity (i.e. based on the ground truth, we see that sentence pair 2 should be significantly closer in meaning than that of sentence pair 1, but the model without LSTM outputs relatively similar scores).

Sentence 1	**Sentence 2**	**Ground Truth**	**w/o LSTM**	**w LSTM**
two dogs running in the snow	A dog laying in the snow.	2.2	0.463	0.635
Someone slices a bottle with a sword.	A man breaks a bottle with a sword.	4.0	0.481	0.845

Table 5: Analysing Short Sentences STS

Similarly, it is hypothesised that the main benefit that pair encoding provides to the PARA task would likely be in the prediction of longer sentences - likely attributed to the fact that by pair encoding

sentences using the BERT tokenizer, BERT would be able to create a joint input representation which may implicitly contain some information with regards to how these two sentences are related.

Given that the above qualitative analysis was done only on a select number of sentence pairs, the insights derived with regards to the relative performance of the model on long and short sentence pairs might not necessarily generalise completely to the full data. Nonetheless, this analysis does provide a plausible reasoning as why and how the incorporation of LSTM into the downstream model for STS and use of sentence pair encoding for PARA significantly helps performance.

5.2 Frequency Embedding and Filtering

In our pursuit to enhance model performance through signal processing techniques, we conducted experiments incorporating Fourier Transform (FT) and Discrete Cosine Transform (DCT) into our minBERT-based models. Initially, we applied FT and DCT directly after minBERT embeddings, aiming to convert these embeddings into their frequency domain representations. However, this approach required significantly more epochs to match the performance achieved by a simpler baseline at just 10 epochs. To address this, we refined our strategy by applying DCT to convert embeddings into the frequency domain, multiplying by a filter array composed of floating-point numbers ranging from 0 to 1, and then inversely transforming them back to the time domain before passing them to downstream tasks. This method, inspired by signal processing's noise reduction techniques, aimed to diminish less relevant frequency information and align the model's performance with our simple baseline (see experiment (3) in Table 2). The training process and performance of each approach in this section were logged in wandb (See Figure 5).

Further experiments isolated the application of DCT-filter-inverse DCT combination to individual downstream tasks—PARA, STS, and SST (see experiment (4), (6), (5) in Table 2). While these task-specific applications did not significantly outperform our best model, applying the combination to the PARA task yielded the highest overall dev score, suggesting a nuanced response to frequency-focused modifications across tasks. However, extending this combination to all tasks concurrently did not surpass the performance of task-specific applications(see experiment (7) in Table 2), indicating a complex relationship between frequency domain processing and task-specific modeling.

From Table 2, the experiment (7), which employed the DCT-filter-inverse DCT combination for all tasks reached the highest accuracy for the paraphrase detection task. To further elucidate the effectiveness, we constructed the confusion matrices for both DCT-enhanced model, exp (7), and the model from experiment (2), which reached the highest overall score. We compared their predictions against the ground truth for the dev set. The confusion matrix analysis revealed that the DCT-enhanced model has the true positive rate of 30.79%, the true negative rate of 55.77%, the false positive rate of 6.7%, and the false negative rate of 6.74%. The prediction results from model in experiment (2) has the true positive rate of 30.76%, the true negative rate of 55.53%, the false positive rate of 6.94%, and the false negative rate of 6.76%. The improvement in reducing incorrect classifications - higher true positive rate and true negative rate for the DCT-enhanced model - suggests that the application of frequency filtering could be helpful for diminishing non-essential information and capturing structural clues during paraphrase detection. By focusing on more relevant frequency components, the filter appears to refine the model's ability to discern paraphrase more accurately.

The bar chart in Figure 6 illustrates the distribution of prediction scores across the dev set, comparing the ground truth with the outcomes of model(2) and model(6) as presented in Table 2 for the Sentiment Analysis task. Model (2), which achieves the highest overall score on the dev set, tends to predict score of 1 and 3. In contrast, model (6), which excels with the highest accuracy in this task, presents a more balanced prediction distribution across all scores. Given that the DCT approach was not directly applied to the SST downstream model in (6), it was hence unexpected that the model's performance on SST improved. We hypothesize that could be a result of differing model gradient directions, so changes to a particular downstream head might influence the performance of other heads. More experiments to investigate this are warranted before a definitive conclusion can be drawn. Nonetheless, we still believe that the DCT-filter-inverse DCT combination has its potential to enhance the model's ability by selectively emphasizing salient features in the frequency domain.

5.3 POS Tag

As documented, the addition of POS tags decreased the model's performance on the downstream sentiment analysis class marginally (-0.008). This was an unexpected observation given that POS tags have shown to be a useful input feature to machine learning models (that do not utilise BERT) performing sentiment classification. A plausible reason for this unexpected worse performance could be attributed to the fact that BERT embeddings obtained from pre-trained weights are rich and hence might already capture most of the useful syntactic information from the input sentence. In a bid to analyse the plausibility of the reasoning, we look at two sentences that we initially though would perform better with POS tags (Table 6). For both sentences, we had hoped that by including POS tags as input feature, the downstream model for the SST task would be able to recognise that the lack of adverbs like "very" in the sentence, which connote stronger intensities of sentiment, might imply that the sentence was more likely within the "neutral" ranges as opposed to being positive. While POS tags didn't correct the classification of the first sentence, it helped with that of the second one.

Sentence	**Ground Truth**	**w/o POS**	**w POS**
It 's a lovely film with lovely performances by Buy and Accorsi.	3	4	4
Entertains by providing good , lively company .	3	4	3

Table 6: Analysing the effects of adding POS tags

This marginal specific improvement could have come at the cost of increased model complexity (due to the additional learnable parameters (NN Embedding) to convert the one-hot POS tags from spaCy to dense representation), which is what we hypothesised to have hurt overall model performance.

5.4 Best Model on Test Set

Based on the various experiments conducted, we have decided that the model with LSTM and pair encoding yields the best performance. On the test leaderboard, it attained a overall test score of 0.767 which is inline with the overall dev score of 0.767 attained.

6 Conclusion

This paper first explored the extent to which a combination of commonly adopted downstream task-specific model layers and procedures help to improve model performance before proceeding with an in depth investigation into whether the incorporation of novel methods such as frequency filtering and language features in downstream models can lead to further improvements on the SST, PARA and STS tasks. Our results have illustrated that: (i) the incorporation Pair Encoding and LSTM into the downstream models leads to significant improvement in overall score (+0.12) relative to the simple baseline, but (ii) the effects of other extensions - especially those of DCT - on overall model performance are less clear. Nonetheless, it is worth noting that while our DCT models did not yield the highest overall dev scores, they have yielded superior performances over the LSTM + Pair Encoding model in selected subtasks, illustrating the promise of this approach.

We recognise that there are - however - certain limitations of our current work that can hopefully be addressed in the future. The primary limitation being that our multi-task fine-tuning did not take into account the fact that gradients directions of different tasks may conflict. This is likely what resulted in our observations that a change to a select downstream model head influenced the results of other model heads - especially prevalent in the runs involving DCT. We believe that this problem can be resolved in the future with an in-depth exploration into gradient surgery(Yu et al., 2020). Another limitation identified is that we were unable to use the full PARA train dataset due to system memory limitations. An exploration into how to better merge training datasets with vastly differing training set sizes is warranted to unlock the full potential of the provided datasets. Some other avenues for future work that we are interested in would be seeing if and how hyper-parameter tuning will enable us to explore the full potential of our model as well as testing the performance limits of a pretrained model (with frozen embeddings) combined with downstream models.

Team Contributions: Joint equal effort. Benita: LSTM, postags models. Tina: DCT models. Mutual cross-checks, with model refinements and iterations done together.

References

Qiwei Bi, Jian Li, Lifeng Shang, Xin Jiang, Qun Liu, and Hanfang Yang. 2022. Mtrec: Multi-task learning over bert for news recommendation. In *Findings*.

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.

Krishnamurthy Kalaivani, R. Felicia Grace, M. Aarthi, and M. Boobeash. 2018. Classification of sentiment reviews using pos based machine learning approach. *International journal of engineering research and technology*, 6.

James Lee-Thorp, Joshua Ainslie, Ilya Eckstein, and Santiago Ontanon. 2022. Fnet: Mixing tokens with fourier transforms.

Jonas Mueller and Aditya Thyagarajan. 2016. Siamese recurrent architectures for learning sentence similarity. *Proceedings of the AAAI Conference on Artificial Intelligence*, 30(1).

Elvys Linhares Pontes, Stéphane Huet, Andréa Carneiro Linhares, and Juan-Manuel Torres-Moreno. 2018. Predicting the semantic textual similarity with siamese CNN and LSTM. *CoRR*, abs/1810.10641.

Alex Tamkin, Dan Jurafsky, and Noah D. Goodman. 2020. Language through a prism: A spectral approach for multiscale language representations. *CoRR*, abs/2011.04823.

Eka Surya Usop, R. Rizal Isnanto, and Retno Kusumaningrum. 2017. Part of speech features for sentiment classification based on latent dirichlet allocation. In *2017 4th International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE)*, pages 31–34.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. 2020. Gradient surgery for multi-task learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 5824–5836. Curran Associates, Inc.

A Appendix (optional)

A.1 Figures

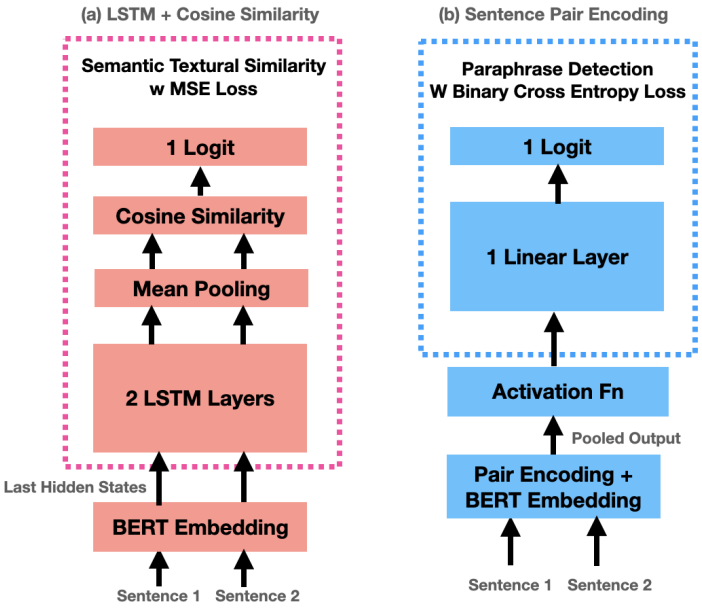

Figure 1: Incorporation of LSTM and Pair Encoding

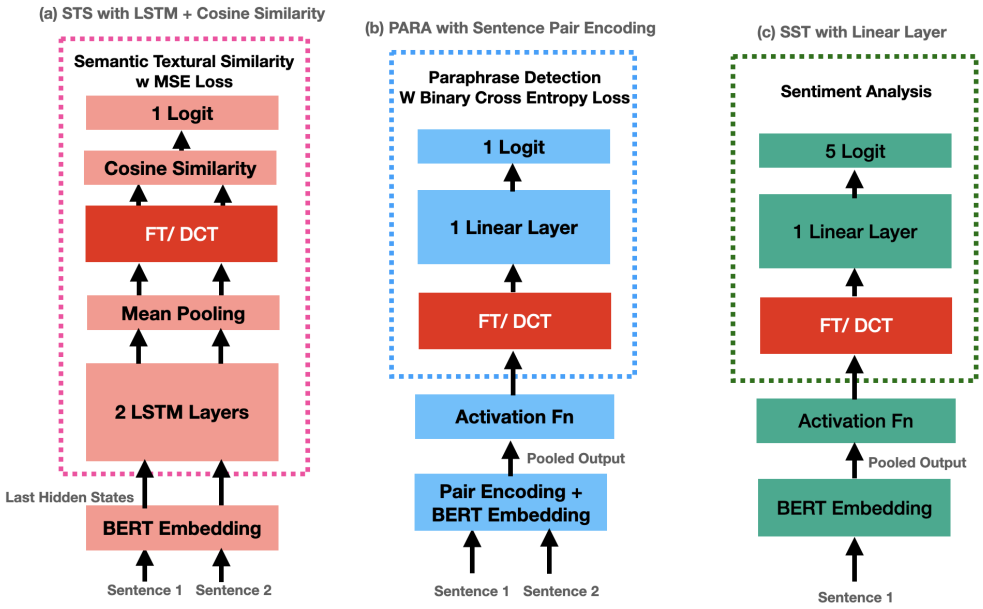

Figure 2: Incorporation of the FT/DCT layer for three downstream tasks

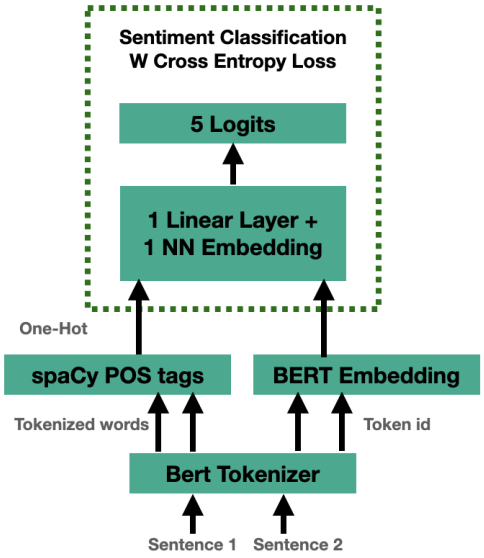

Figure 3: Incorporation POS tags

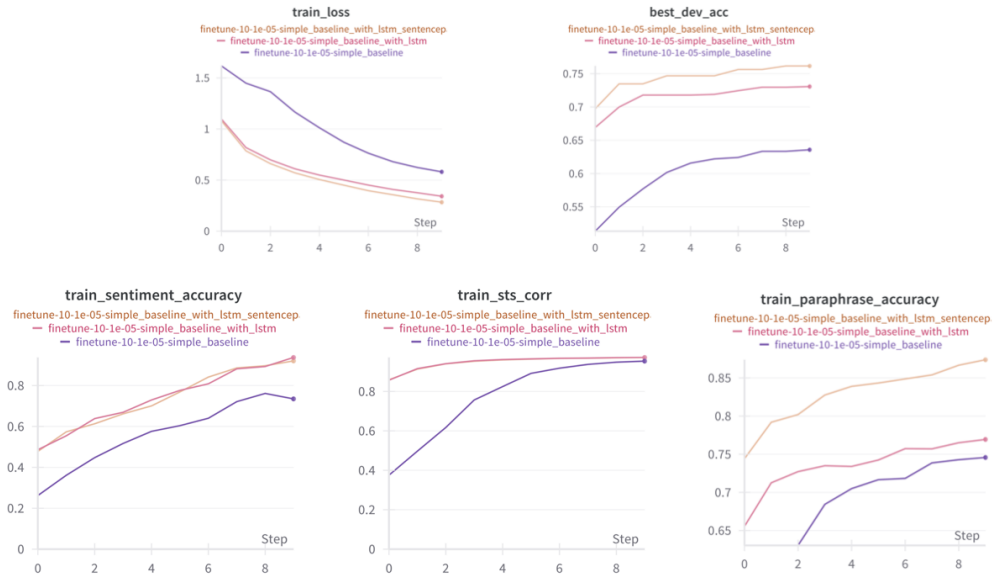

Figure 4: Training Curves for simple baseline and extensions

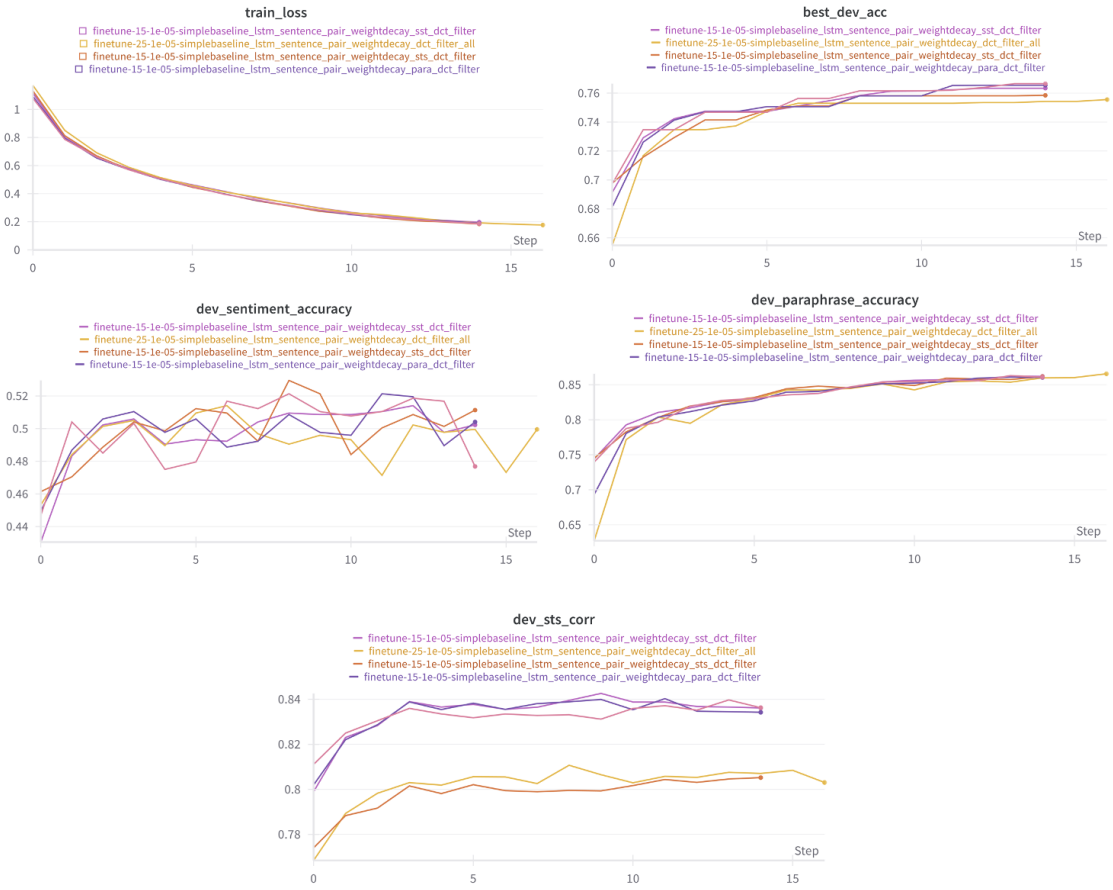

Figure 5: Training Curves for DCT approach and performance on dev set

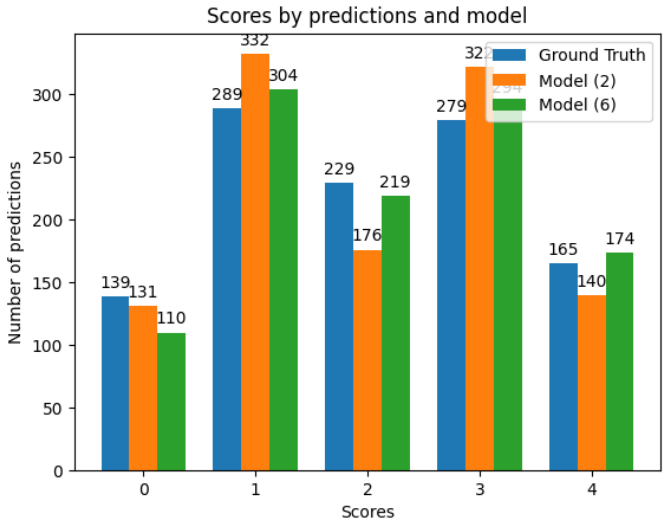

Figure 6: Distribution of prediction of score for dev set from ground truth, model(2), and model(6) in Table 2 for Sentiment Analysis task