CSE 354 Final Project: A Comparison of Financial Sentiment Analysis Performance Using Pretrained DistilBERT and ELECTRA Models

Sahibjot Bhullar:113294299, Ryan Engel: 113150597,
Ramy Abdulazziz:107517524

April 2023

1 Project Overview

1.1 Overall Goal:

For this project, the main goal was to train different models to predict sentiment analysis of financial documents, finding the best type of model, training data, training routines, and hyper-parameters to do so. We begin by comparing the general accuracy of pre-trained models on the Financial Phrasebank dataset, and through experimentation and comparison, determine the best model to use for more general sentiment analysis. Overall, we wish to provide a model that is able to accurately determine the sentiment of any general financial document or article. More formally we train our models to do multi-class sentiment analysis, aiming to classify documents into several sentiment classes (positive, negative, neutral). We formalize this as a multi-class classification problem where our model can be represented as a function that takes in a document D as input and produces a sentiment score S as output:

$$f : D \rightarrow \{pos, neg, neutral\}, f(D) = S$$

1.2 Possible Challenges:

The task of determining sentiment analysis of financial documents is difficult because there is not only quantitative data to consider, but there are multiple textual features like social media posts and speeches that can have drastic influence on the movement of stocks. A lot of existing pre-trained models can not accurately be used in

finance and economics because financial language is different from language that is used everyday. For example the statement "interest rates are rising" does not have any negative words, and because of this it would not be considered negative by most NLP models. Another potential challenge may be that the datasets themselves could also be highly imbalanced, with unequal numbers of each sentiment class. This is the case for the Financial Phrasebank dataset, which is split into datasets that have different levels of consensus. This shows that finding high-quality labeled data is incredibly difficult in this domain, and could potentially create bias within the models trained on these datasets. Due to these reasons predicting sentiment analysis of financial documents will certainly be a challenging task. The paper, "WHEN FLUE MEETS FLANG: Benchmarks and Large Pre-trained Language Model for Financial Domain" discusses how NLP models are used in the finance domain and also mentions some of the challenges we discussed above. The link to this paper can be found here: `https://cs.stanford.edu/~diyiy/docs/emnlp_flang_2022.pdf` Although challenging, this paper shows promising results for using both the DistillBERT model and the ELECTRA model to predict sentiment analysis of financial documents.

1.3 Solution Strategies:

Some ideas that we thought about trying were to fine-tune a pre-trained model like ELECTRA or DistilBERT. Pre-trained models are language models that are trained on an unsupervised language modeling task. Using a pre-trained model can be efficient in time and space complexity, because this data can be easily trained on google colab, which is where we wrote the code for this project. This was a good start because we were able to compare the results of both models to get a good idea of how they were performing. Another idea that we had was to analyze few-shot performance on large language models. We decided to do this after fine-tuning the pre-trained models. To perform few-shot learning, we used the fine-tuned models to generate predictions for new unlabeled data. The unlabeled data for our few-shot experiment was a set of separate news articles that were given a sentiment value based on the judgment of our group. Our goal for this idea was to maximize the few-shot performance of both the DistilBERT model and the ELECTRA model by fine-tuning these models on the custom dataset.

1.4 Evaluation and Analysis:

To evaluate our fine-tuned models and our few-shot performance, we trained our models on different layers which included frozen layers, top 2 training, top 4 training, and all training to figure out which

layers of our model would lead to the best results. We plotted the accuracy, recall, F1 score and loss to measure our models overall accuracy. We used the $F1$ score to get a good idea of how each model compares, by taking the harmonic mean of precision and recall from each experiment. We calculated the $F1$ score by using the formula shown below. To check if our models could predict sentiment analysis of new financial documents, we compared the results of each models few-shot performance with the groups overall consensus of each document in the custom dataset. We then decided the accuracy of these models based on if the output was the same as ours.

$$F1\ score = 2 \cdot \frac{(precision \cdot recall)}{(precision + recall)}$$

1.5 Summary of Results:

After performing experiments and obtaining their results, we found that the accuracy of many of our tests were similar and in some cases higher than the accuracy achieved in the paper cited above. We observed that we were able to fine-tune these models to achieve sentiment analysis of documents with reasonable accuracy considering the training data that we had access to. Overall, both models were very similar in accuracy, and by comparing the F1 scores to get a good idea of each models results, we concluded that the DistilBERT model was slightly more effective than the ELECTRA model. We believe that the reason for this was because we were only using the ELECTRA-small model throughout this project. We tried using the ELECTRA-large model however we faced GPU limits in our google colab notebook, we also believe that if we were able to use the ELECTRA-large model then this would have significantly better performance than the DistilBERT model because of its larger dataset. When it comes to the few-shot performance, we found that both models were fairly accurate when predicting sentiment analysis of the news articles in the custom dataset. Something that we could have done better for this idea was to include more data, we used 11 articles from yahoo finance and observed the accuracy of our models on these 11 articles. Our group observed that on the few-shot performance of the models, the DistilBERT model was much more effective in predicting sentiment analysis of the news articles than the ELECTRA model.

2 Ideas

2.1 Fine-Tuning of Pre-Trained Models:

Our first idea was to fine-tune pre-trained NLP models. Fine-tuning a pre-trained model involves taking a base model that has already been trained on a large dataset and adapting it to a specific task or domain with a smaller dataset. This process leverages the pre-trained model's ability to extract general features, while fine-tuning allows the model to learn task-specific patterns and characteristics from the smaller dataset. This technique typically requires less data and training time compared to training a model from scratch, as the pre-trained model has already captured general patterns, making it a practical and efficient approach for achieving better performance on various NLP tasks. For this experiment the models that we used were DistilBERT and ELECTRA, we evaluated how these models compared and focused on achieving the highest possible accuracy between both models in our experiments.

2.2 Few-Shot Performance:

Our second idea was to use few-shot performance on pre-trained models. This technique leverages the extensive knowledge and patterns captured by the models during pre-training on a diverse range of texts. When given a small set of examples, the models can generalize from these examples to perform the desired task effectively. This approach demonstrates the potential for large language models to tackle a wide array of NLP tasks, without requiring explicit fine-tuning or a task-specific dataset. We used the few-shot performance technique on both the DistillBERT model and the ELECTRA model to compare their performance. Likewise, we tested these models using different layers in order to understand the most effective way to evaluate these models and their performance.

3 Experimental Setup

3.1 Models:

The models we chose to use were the DistilBERT model and the ELECTRA model. The DistilBERT and ELECTRA models are both transformer-based models, they are not recurrent neural networks. DistilBERT is a smaller, more efficient version of the BERT model. It is created by applying knowledge distillation, a process where a compact model is trained to mimic the behavior of a larger, more complex model. DistilBERT retains most of the original BERT model's performance while having fewer layers and parameters, thus requiring less

computational resources and enabling faster inference. DistilBERT typically has fewer layers than BERT, and BERT-base has 12 layers. DistilBERT is a pre-trained model, and it can be further fine-tuned for specific tasks. ELECTRA (Efficiently Learning an Encoder that Classifies Token Replacements Accurately) is another transformer-based model designed for efficient pre-training. It employs a novel pre-training approach, using a generator-discriminator architecture in which the generator proposes masked token replacements, and the discriminator predicts whether each token in the input sequence is original or replaced. This method allows ELECTRA to learn from all input tokens rather than only masked ones, as in BERT. ELECTRA-small, for instance, has 12 layers, while larger variants may have more layers. ELECTRA is also pre-trained and can be fine-tuned for specific tasks.

3.2 Dataset:

The dataset used was the Financial Phrasebank dataset, split into three subsets for training, validation, and testing. The dataset contains sentences labeled with one of three classes $C \in positive, negative, neutral$. In total, the dataset contains 4,840 phrases specific to the financial domain, comprised of sentences from financial news labeled by sentiment. The dataset is divided by the agreement rate of the 5-8 annotators. For our experiments, our training set was the $sentances\_75agree$ split, our validation set was the $sentances\_66agree$ split, and our testing set was the $sentances\_all\_agree$ split. We decided to divide the dataset in this way in order to let the model train on the most amount of data possible. The 75% split we felt provided the best balance between corpus size and accurate labeling. Similarly, we felt the $sentances\_all\_agree$ split would provide the best testing set due to the agreement of all annotators. For our first idea, the training instance was the $sentances\_75agree$ subset and the test instance was all of the data in the Financial Phrasebank. For our second idea, our training instances was still the sentences75agree subset but this time our test instance was a custom dataset of new news articles chosen by our group. The articles of this custom dataset consisted of 3 neutral, 4 negative, and 4 positive, making a total of 11 articles.

3.3 Evaluation Metrics:

For our first idea, fine-tuning the pre-trained models, we used automatic evaluation metrics. Specifically the metrics we analyzed were the loss, precision, recall, and F1 scores. The loss is a measure of the discrepancy between the model's predictions and the true labels. It quantifies the error made by the model during training or evaluation. A lower loss value indicates better performance, and the goal dur-

ing training is to minimize the loss function. Precision measures the proportion of true positive predictions among all positive predictions made by the model. High precision indicates that the model is accurately identifying positive cases and has fewer false positives. Recall measures the proportion of true positive predictions among all actual positive instances. High recall indicates that the model is effectively identifying most positive cases and has fewer false negatives. The F1 score is the harmonic mean of precision and recall, combining both metrics into a single value. It balances the trade-off between precision and recall, making it especially useful when dealing with imbalanced datasets. The higher the value of the F1 score, the better the overall performance is. This metric is particularly helpful when you want to assess the model's performance without favoring either precision or recall. For our second idea, using few-shot performance on the pre-trained models, we used human evaluation of each article. The way our group chose to evaluate the result was to have all group members read each article, and come to a general consensus on whether the article was positive, negative, or neutral. Then we considered what our model's output was and compared the result of each article with the general consensus of the group.

4 Results

4.1 Fine-Tuning of Pre-Trained Models

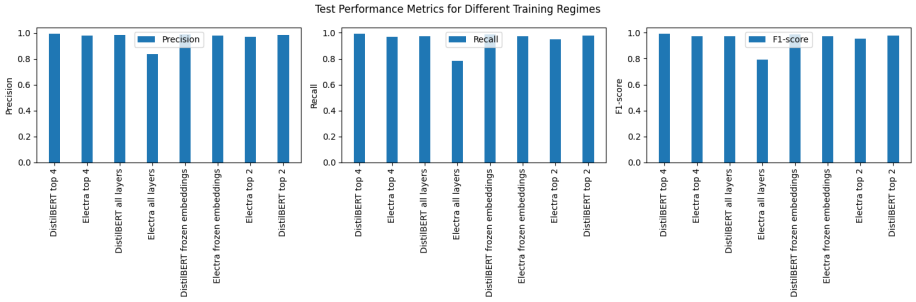

Figure 1: Test Metrics for all models and training regimens on Financial Phrasebank test set

For our first idea, we fine-tuned the DistilBERT and ELECTRA models to obtain the most accurate results. We tested and trained the models using multiple different layer training regimens. Here we provide tables of all the data obtained from testing on frozen embeddings, top 2 layers, top 4 layers, and all layers. The graphs above show the overall performance of the different training regiments that

Model and Training Regime	Precision	Recall	F1 Score
DistilBERT top 4	0.9924	0.9908	0.9906
Electra top 4	0.9816	0.9701	0.9716
DistilBERT all layers	0.9830	0.9740	0.9758
Electra all layers	0.8394	0.7843	0.7913
DistilBERT frozen embeddings	0.9917	0.9890	0.9887
Electra frozen embeddings	0.9810	0.9758	0.9754
Electra top 2	0.9693	0.9520	0.9557
DistilBERT top 2	0.9847	0.9798	0.9795

Table 1: Test metrics for all models and training regimens on Financial Phrasebank test set

were tested. It is worth taking note that these values were significantly higher than the results achieved in the paper cited above. We think this may be because we are training on the sentence75agree subset and testing on the entire Financial Phrasebank dataset.

4.2 Few-shot performance

Model	Accuracy	Precision	Recall	F1 Score
DistilBERT Top 2	0.88	0.92	0.88	0.88
DistilBERT Top 4	0.88	0.92	0.88	0.88
DistilBERT Frozen	0.88	0.92	0.88	0.88
DistilBERT All Layers	0.25	0.06	0.25	0.10
Electra Top 2	0.88	0.92	0.88	0.88
Electra Top 4	0.38	0.45	0.38	0.36
Electra Frozen	0.50	0.75	0.50	0.52
Electra All Layers	0.88	0.92	0.88	0.88

Table 2: Test metrics for all models and training regimens with low amount of neutrals in test set

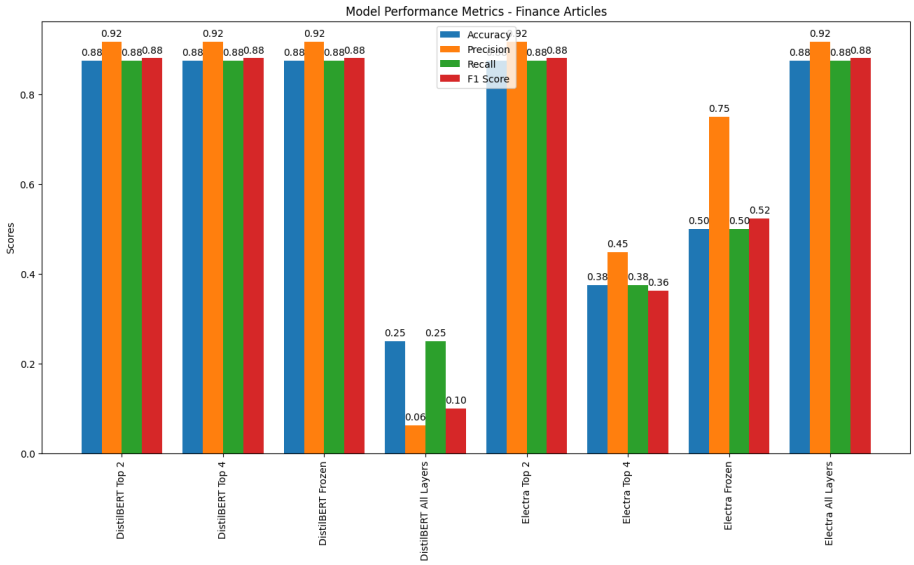

Figure 2: Test metrics for all models and training regimens on finance articles with low amount of neutral data

Model	**Accuracy**	**Precision**	**Recall**	**F1 Score**
DistilBERT Top 2	0.73	0.57	0.73	0.63
DistilBERT Top 4	1.00	1.00	1.00	1.00
DistilBERT Frozen	1.00	1.00	1.00	1.00
DistilBERT All Layers	0.45	0.42	0.45	0.34
Electra Top 2	0.73	0.78	0.73	0.70
Electra Top 4	0.64	0.74	0.64	0.63
Electra Frozen	0.73	0.79	0.73	0.72
Electra All Layers	1.00	1.00	1.00	1.00

Table 3: Test metrics for all models and training regimens on finance articles with high amount of neutral data

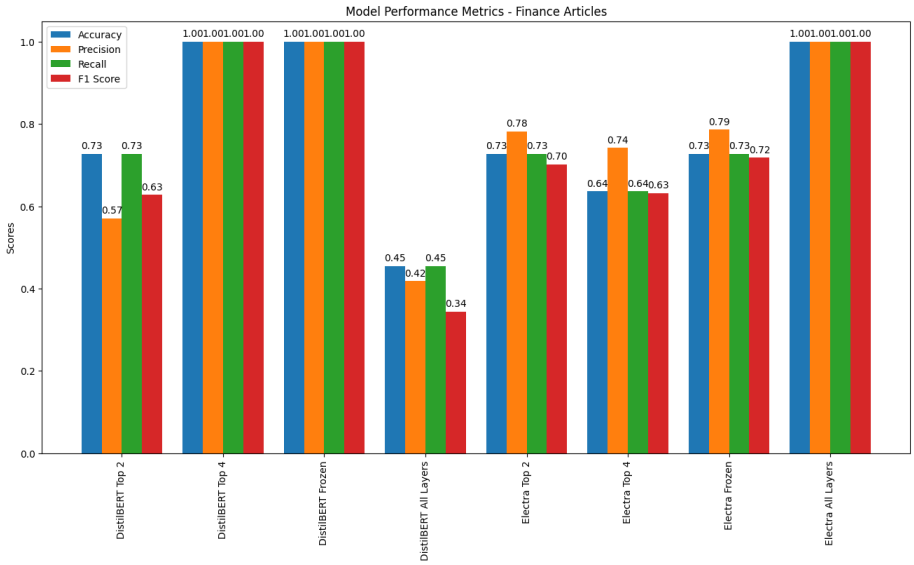

Figure 3: Test metrics for all models and training regimens on finance articles with high amount of neutral data

For our second idea, we used few-shot performance on DistilBERT and ELECTRA to compare the results of these models on a small set of examples. These examples were hand picked financial news articles that were chosen to help emphasize the differences in the results of each model. We ran the tests once with a low amount of neutral data, and then ran the tests again with high neutral data. We did this to try and understand how our models reacted to neutral articles. As you can see from the graphs, the models tested with low neutral data had more consistent results, but the models tested on high neutral data had more inconsistent results with some values that were much higher, and some much lower. We think this is due to the fact that if our dataset is made up of unequal amounts of data, then this could introduce biases into our model. When we tested the model with a dataset that had a low number of neutral articles this introduced bias, thus making the results inaccurate.

4.3 Retraining Few-Shot Performance

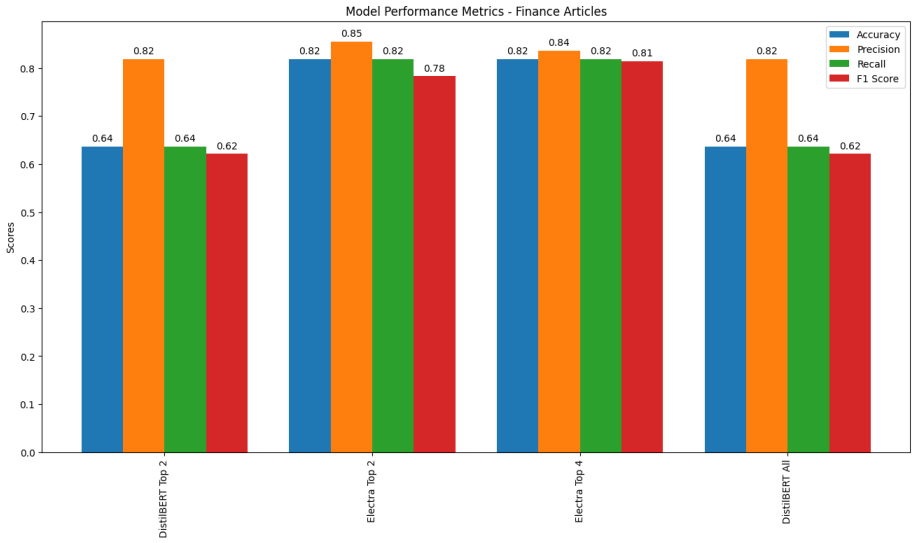

Figure 4: Test metrics for all models and training regimens on finance articles after retraining

Model	Accuracy	Precision	Recall	F1 Score
DistilBERT Top 2	0.64 (+12.33%)	0.82 (+44.74%)	0.64 (-12.33%)	0.62 (+1.59%)
Electra Top 2	0.82 (+12.33%)	0.85 (+8.97%)	0.82 (+12.33%)	0.78 (-11.43%)
Electra Top 4	0.82 (+18.00%)	0.84 (+0.00%)	0.82 (+0.00%)	0.81 (+28.57%)
DistilBERT All	0.83 (+42.22%)	0.86 (+95.24%)	0.83 (+42.22%)	0.90 (+82.35%)

Table 4: Percent change of models after retraining

After initial few shot testing, we took the worst performing models namely, the DistilBERT models with top2 and all layers training, and the ELECTRA models with top 2 and top 4 layer training, and attempted to increase overall performance by retraining for the specific task of analyzing finance articles. The models were allowed to train on the previous testing set, and a new testing set was used to measure overall performance. This led to an overall substantial increase in performance in every model, most notable the DistilBERT all layer model which saw an increase of 42.22% **in accuracy, and a** 95.24% **increase in precision.**

5 Analysis and Discussion

5.1 Failures of the Model:

One failure of our model was that it was very hard for it to classify neutral articles. One reason for this might be because the articles were not just one sentence like our training data. To elaborate, the wording of financial documents always includes some parts of positive and negative sentiment, even when it was neutral. So our models sometimes classified the articles with neutral sentiment as either positive or negative because it was not able to understand how multiple sentences are combined to become neutral.

5.2 Hypotheses:

One hypothesis we made was that neutral articles may be causing the model's failure because the combination of both positive and negative words in a neutral article could convince the model otherwise. Another hypothesis is that financial language has a lot of ambiguity. For example, the sentence "interest rate increases" can be positive or negative depending on the situation. It is helpful for one party and a downfall for the other. A hypothesis we made to describe our model's success is that our model will succeed in predicting strongly negative and positive sentiment articles because those articles will have keywords with high probabilities associated with a specific label, so the model will accurately predict the correct sentiment.

5.3 Examples of Input and Output of the Model:

An example of our first hypothesis is when an article discusses a financial subject and presents all its information, positive or negative, but with a neutral tone. Our model may incorrectly predict it as positive or negative instead of neutral because the article will have keywords associated with positive and negative sentiment keywords. For example, an article on Robinhood might discuss how it is very user-friendly and might help a lot of people to start investing and making profits, but it may later talk about the negatives of Robinhood like people with limited knowledge of options trading can loose a lot of money because Robinhood makes it very easy to trade without the user having vast knowledge on trading. Overall the article is neutral because it is an informative article, but it uses words and phrases such as profits and losing money which can make the model predict the article to be either positive or negative incorrectly.

An example of our second hypothesis is when an article talks about a topic with ambiguity. For example, a high-interest rate is sometimes

positive for the economy or negative depending on multiple factors. When there is inflation, a high-interest rate can effectively control inflation, but sometimes it can cause slow economic growth because it reduces consumer spending and business investment. Sometimes financial statements can benefit one party while affecting another. Another example could be how the housing market sometimes helps the seller or the buyer in different circumstances. Because of these reasons, our model has trouble predicting ambiguous articles correctly.

An example of our final hypothesis could be a strongly positive or negative document. For example, a document talking about recession causing tech job layoffs, this article will have keywords like Bankruptcy, Market crash, recession, layoffs, etc. These words will have a higher probability of being associated with negative sentiment. So our model will accurately predict it as negative or positive based on the context of that article.

5.4 Results of the Hypotheses:

After doing experiments on multiple distinct articles, we found out that our hypothesis was correct. In our hypothesis, we predicted our model would have a problem predicting ambiguous articles. Hence, we tested it on an article with the title "federal reserve interest rate decision of may 3". The model predicted it as positive even though it was a negative article, because it used words and phrases like increased, highest, and improved, which are words generally associated with positive sentiment. We also tested strongly positive and negative articles, and our model accurately predicted the right sentiment. For example, we tested the article "Housing is so unaffordable that banks are losing money for each mortgage they finance for the first time ever", which had a lot of keywords associated with negative sentiment like losing, unaffordable, lost, and negative profits which made it very easy for our model to predict this as negative. As predicted, when we tested on articles with a neutral sentiment, sometimes it had a problem identifying them as neutral depending on what layers it was trained on and what model we tested it with. An article we used called "There are two types of stocks on Robinhood" talked about the features of Robinhood and what features are available on the app. The article had a neutral sentiment, but because it used words like beneficial, profits, and invest our models labeled the article as having a positive statement.

6 Learning Outcomes

In general, both DistilBERT and Electra models have good generalizing ability, showing the overall effectiveness of the pre-trained models word embeddings. However, since these models are pre-trained on a more general dataset, in certain cases, their ability to accurately classify financial documents are affected. The Financial Phrasebank dataset has text that is extremely specific to finance, and this domain specific language can change performance based on the type of training that is used, and the models ability to adapt to text in the financial domain. We see this being further exacerbated when comparing the overall results for few-shot performance on various finance articles.

From our results, we observed that DistilBERT and ELECTRA models, when fine-tuned with different layer training regiments, achieved significantly higher results compared to the baseline cited in the original paper. Among these, the DistilBERT model with top 4 layers training provided the best performance in terms of precision, recall, and F1 score.

The data however, is not completely straightforward, as during few-shot performance analysis the addition of large amounts of neutral sentiment data resulted in somewhat inconsistent results. This highlights the necessity to ensure an equal distribution of data, during testing, training, and validation. This ensures, that no bias is introduced into the model affecting its accuracy.

Overall, the experiments highlight the ability to effectively fine-tune generalized pre-trained models such as DistilBERT and ELECTRA on even domain-specific tasks. Although as shown, the type of training regiment used, along with the overall distribution of data, caused significant changes in overall performance. Based on our results, we conclude that it is imperative to carefully select fine-tuning strategies and training regiments in order to achieve optimal performance in extremely domain-specific tasks, such as financial sentiment analysis.

To this end, we conclude that it is also extremely important to keep in mind model architecture when fine-tuning for domain specific tasks. Our initial results for few-shot analysis of financial articles highlight these underlying differences. Both DistilBERT and ELECTRA have different pre-training objectives that result in different levels of adaptability and robustness during experimentation. Specifically, we can see that the ELECTRA model during all layer training, has a significantly better ability to detect nuances in input data, as compared to

the DistilBERT all layer model. DistilBERT in this case shows excellent generalizing ability, and it is most likely the case that having fewer layers and parameters than the full sized BERT model does not have the capacity to capture nuances and complexity during this type of training. As such, when working with pre-trained models that have smaller architectures, the results strongly suggest that best overall performance occurs from not retraining word embeddings. The comparison between the different models show that model architecture is always an especially important consideration for domain specific tasks.

7 Contributions

All group members contributed equal amounts of work, where Ramy did most of the programming, Sahibjot and Ryan both helped with programming and wrote most of the paper. All group members shared creative ideas and opinions about the project, and consulted each other with questions to further advance our understanding of the material. All of the problems and challenges faced throughout the project were discussed and handled as a team, and we were all satisfied with the results of our work.

8 Code and Resources

- A link to our google colab notebook can be found here: `https://colab.research.google.com/drive/13rYAWHDp4By29AVFcqol0kkmpOtq1wRz?usp=sharing`
- Financial Phrasebank Dataset: `https://huggingface.co/datasets/financial_phrasebank`
- Trained Models: `https://drive.google.com/drive/folders/1-1ea4F-T49fFs719-doudem9adyQUc7O?usp=sharing`
- Research Paper: `https://cs.stanford.edu/~diyiy/docs/emnlp_flang_2022.pdf`
- Github Repo With Detailed README: `https://github.com/Ramy-Abdulazziz/CSE354-Final-Project`
- Software Requirements:
  - Python 3.x installed
  - PyTorch installed
  - Transformers library installed
  - Datasets library installed

  - Beautifulsoup4 library installed
  - Responses library installed
  - Scikit-learn library installed
  - Matplotlib library installed

- Acknowledgments: For this project we reused portions of code from CSE 354 HW Assignment 3, provided to us by Professor Niranjan Balasubramanian, and the CSE 354 graduate TA'S. The Training and Dataloader classes were used and modified to accommodate different aspects of the training and testing. The testing class was also modified to handle our specific test requirements. These changes involved modifications to use the dataset loaded via the dataset library as well as our custom dataset (articles from Yahoo Finance), Tokenizing our custom dataset, handle both the DistilBERT and ELECTRA models, and add logging/graphing to visualize our results.