Predicting Hedge Fund Holdings from 10K/Q text Analysis

Stanford CS224N {Custom} Project

Oussama Fadil
Department of Computer Science
Stanford University
fadil@stanford.edu

Abstract

We present a NLP for model for applications to financial data and sentiment classification in particular. We reach a 96% accuracy on the Financial PhraseBank dataset, which is above benchmarks for the corpus. Our broader goal is to predict hedge fund holdings based on text analysis of financial data. To this end, we select 50 random tickers from the S&P and show a positive correlation between sentiment predicted by the model on earnings call transcripts and aggregate position changes in the subsequent quarter.

1 Key Information to include

- External collaborators (if you have any): None
- External mentor (if you have any): None
- Sharing project: CS320/230

2 Introduction

Instead of trying to predict stock market returns or performance of a single stock, we turn our attention to predicting hedge fund holdings. We believe this to be an interesting problem given that funds typically charge hefty fees (2% of assets under management and 20% of profits) for a process that could be predictable. Many reputed and large ($20B+ in AUM) hedge funds do not have any programmers and rely on a process that is manual and research intensive to initiate and size positions. We believe that even partially predicting positions held by funds is immensely valuable.

3 Related Work

Several papers looked at the predictive power of sentiment on stock prices and showed little correlation (e.g. Frisbee et al., [1],[2]). Most of these papers however are dated (pre 2015) and rely on third party datasets for sentiment scores.

A more recent effort by Araci et. al [3] aimed to use cutting edge NLP models for sentiment prediction from financial news. The effort, which resulted in a model called FinBERT, however stopped short of making any predictions on financial asset prices or market movements.

In this sense, our project builds on FinBERT by aiming to improve its accuracy. It also builds on earlier efforts by testing whether Transformer based models can result in more expressive sentiment scores.

4 Approach

We initially started by using a vanilla BERT model[4] from the 'transformers' package[5]. However, we moved to using DistilBERT[6] for our final results. The move allowed us to train faster and hence include more data and reach better overall performance.

DistilBERT is a compressed version of BERT based on a technique called distillation that was popularized by Hinton et. al in 2015[7]. The idea behind distillation is to train a smaller model (DistilBERT) on the output of a larger model (BERT). The output of the larger model is treated as a smooth version of the original target labels that the large model was trained on. In our case, where the target output is binary, we use a cross-entropy loss that takes the following form for the original model:

$$\mathcal{L}_{BERT} = \sum_{i}^{n} y_i log(\hat{y}_i)$$

Note that the y_i correspond to the ground truth "hard" labels and the $\hat{y}_i$ to the "smooth" predictions. The corresponding DistilBERT loss is then defined as:

$$\mathcal{L}_{DistilBERT} = \sum_{i}^{n} \hat{y}_i log(\hat{z}_i)$$

where $\hat{z}_i$ correspond to the DistilBERT predictions.

Now that we have defined a loss function for the mode, we have to settle on an architecture for DistilBERT which captures the same ideas, but is sparser. In this regard, we rely on the architecture presented in the DistilBERT paper[6] and implemented in the transformers package [5].

The architecture makes three changes to the original BERT architecture. First, it simplifies the tokenization process by dropping the 'Segment Embeddings' representation (Figure 1). Next, it drops the next sequence prediction output from the model. The NSP output is used by BERT to get a hidden representation of the input sequence during pre-training.

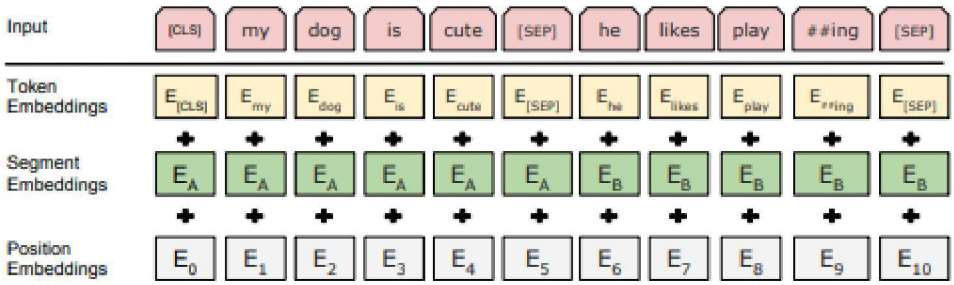

Figure 1: Original BERT Embedding Diagram[4]. DistilBERT removes the middle 'Segment Embeddings' from the input representation

These two changes make it less suitable for tasks such as next sentence prediction or questions answering. However, for our purpose (classification) it is of little importance. In fact, the original BERT authors reported an insignificant (.1%) drop on SST2 when forgoing next sentence prediction (NSP) in training (Figure 2).

The last architectural change consists of dropping half of the hidden transformer layers, which reduces the representational power of the model but also drops the number of parameters by a factor of 2. All these changes are made prior to distillation training.

At train time, we strip the last layer of the model and add a binary classification head to further fine-tune the model on SST2. Next, we fine-tune the model on the Financial PhraseBank data using a three-way classification head.

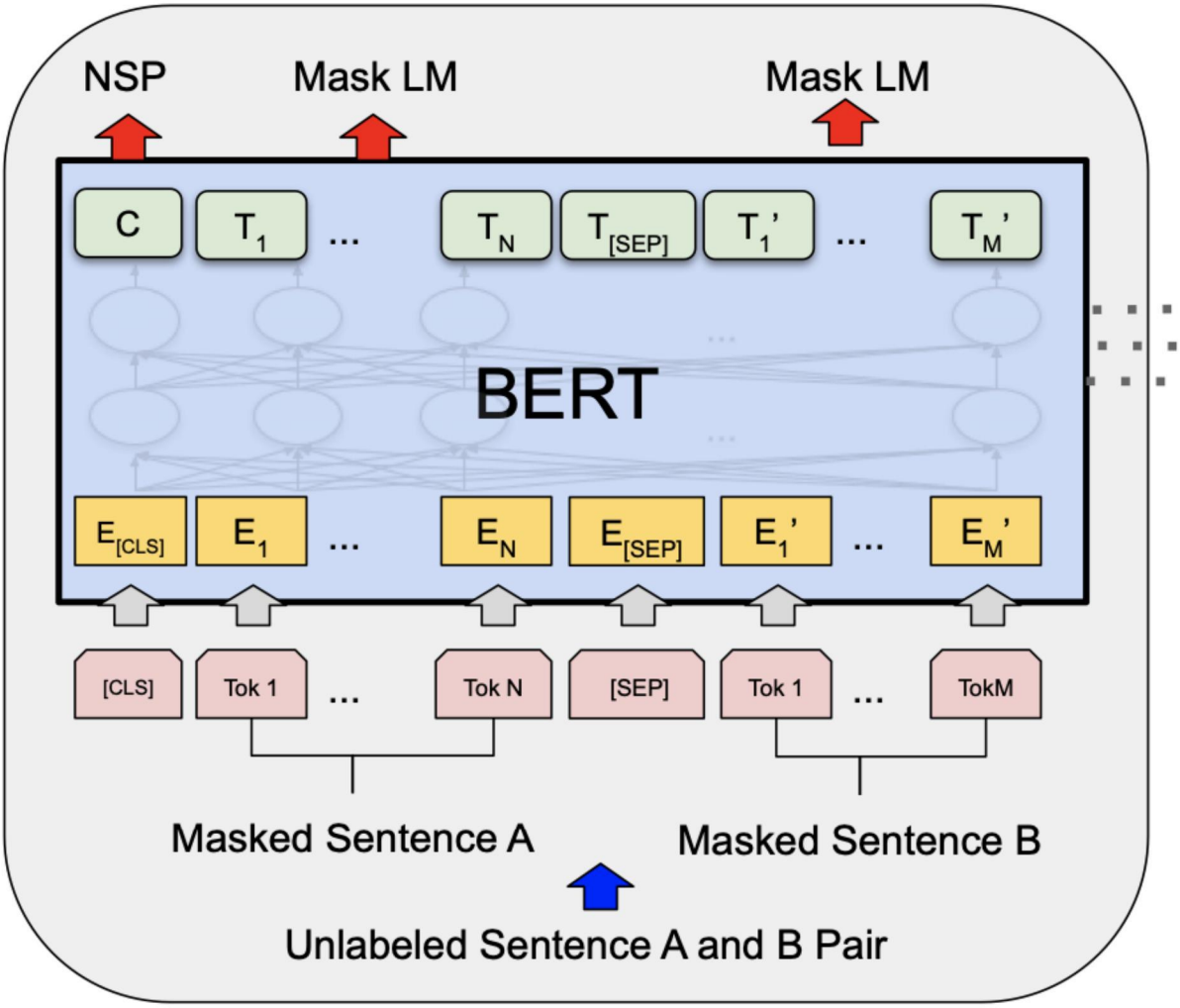

Figure 2: Original BERT Pre-Training Architecture[4]. DistilBERT gets rid of the NSP output in training

The text is tokenized using a pre-trained HuggingFace [8] tokenizer (bert-base-uncased). The tokenizer weights are frozen and not modified during fine-tuning and training. For training data, each sample is tokenized independently.

We then test our model by predicting average sentiment from earning calls. The text transcript of each call is tokenized with each sentence treated independently. The model prediction is then averaged across sentences to yield a single score for each call.

5 Experiments

5.1 Data

We mainly rely on four data sources:

- **Holdings Data**: A database of hedge fund holdings extracted from quarterly 13-F filings spanning from 1990 to 2017. For each quarter, we have a list of fund (identified by CIK) and for each fund, a list of securities (identified by CUSIP) including how many shares of the security were held by the fund at filing[9].

Agreement level	Positive	Negative	Neutral	Count
100%	%25.2	%13.4	%61.4	2262
75% - 99%	%26.6	%9.8	%63.6	1191
66% - 74%	%36.7	%12.3	%50.9	765
50% - 65%	%31.1	%14.4	%54.5	627
All	%28.1	%12.4	%59.4	4845

Figure 3: Summary of Financial Phrasebank data as presented in the FinBERT paper[3]

- **Financial Phrasebank**: A dataset of sentences extracted from financial news[10]. Each sentence is labeled as 'positive', 'negative', or 'neutral' by a group of experts. Not all experts agree on the label, and only the majority label is retained. However, the dataset is separated into four files based on the fraction of experts who agree on the majority label: 50%, 66%, 75%, and 100% (see Figure 3).
- **Earnings Call Transcripts**: Transcripts of quarterly earnings calls where companies go over results for the quarter are also extracted via an API [11] published by 'Financial Modeling Prep'.
- **Company Filings**: Text data extracted from 10K/Q quarterly filings. All filings are found on the Edgar database [12] and extracted using the sec-edgar-downloader package [13].

5.2 Evaluation Method

Our baseline for performance evaluation on the Financial PhraseBank dataset is to compare our validation accuracy (on three way classification) against the results posted in the FinBERT paper [3] (86% validation accuracy). We use a Multiclass Cross Entropy (CE) Loss as an objective when training the model. The model makes a discrete prediction:

- 0: Negative Sentiment
- 1: Neutral Sentiment
- 2: Positive Sentiment

Our approach for predicting holdings between quarters is to feed our model data on each CUSIP separately. This consists primarily earnings call transcripts. The model makes a prediction for each CUSIP (as opposed to each CUSIP/CIK pair) that is a score in the 0 to 3 range.

As implied by our evaluation approach, for each security, positions are aggregated across funds and compiled into how many shares are held by institutional investors in total. We then measure the quarter over quarter change in the number of position held for each CUSIP. We expect to see a high correlation between the latter and the CUSIP's score described in the previous paragraph. We run the evaluation across 50 randomly sampled tickers and choose Q4 of 2016 for our evaluation.

5.3 Experimental Details

We first train on the 50% agreement subset of the Financial Phrasebank. Our goal here is to make sure that the model works with decent hyper-parameter settings. We do not warm-up the weights and train for 5 epochs in total. Our learning rate is set to 2e-5 and weight decay is set to 1e-2. We achieve an accuracy of 88% on the validation dataset, which is slightly above existing benchmarks.

We then move to running a systematic hyper-parameter sweep including the following variable-range combinations:

- Learning Rate: 1e-6 to 1e-2
- Weight Decay: 1e-6 to 3e-1
- Training Epochs: 1 to 10
- Warmup Steps: 0 to 600

- Training Batch Size: One of [2, 4, 8, 16, 32, 64]
- Random Seed: Random integer between 1 and 50

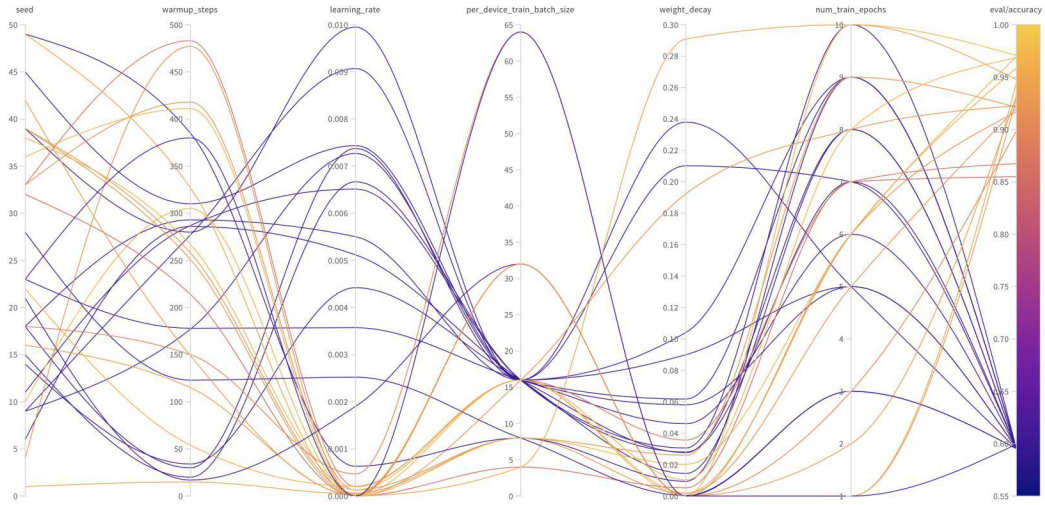

Figure 4: Hyper-parameter sweep results showing final validation accuracy (right) as a function of hyper-parameter values. The main factor seems to be the learning rate, which has to stay under 1e-4 for the model to perform well in terms of validation accuracy

We show the results from our parameter sweep in Figure 4. For our final training run, we choose the following hyper-parameter settings:

- Learning Rate: 1.5e-4
- Weight Decay: 1e-2
- Training Epochs: 8
- Warmup Steps: 280
- Training Batch Size: 16

We believe the most important choices to be the learning rate and weight decay. In fact, looking at the sweep output, these parameters separate runs clearly at 85% accuracy. We settle for a learning rate in the upper range of values that lead to high accuracies. We set our weight decay to its default value of 0.01 given that it doesn't seem to affect accuracy a lot as long as it is set to a reasonable value. The choice of training epochs is based on the observation that we get little performance upside between epoch 8 and 10. Lastly, we choose a value for warm-up steps that is in the middle of the range and not too close to bad runs at <50 warm-up steps.

Once we settle on these parameters, the model is trained on the full corpus of financial data (including all agreement levels). It reaches a validation accuracy of 96% and is saved as our final model.

Lastly, we turn our attention to making predictions on the earnings transcript data.

5.4 Results

Model	Loss	Accuracy	F1 Score
Vanilla BERT	0.38	0.85	0.84
FinBERT-task	0.39	0.86	**0.85**
FinBERT-domain	**0.37**	**0.86**	0.84

Figure 5: Summary of results reported by FinBert authors [3] comparing a Vanilla BERT implementation to their domain specific solution.

We achieve a higher performance than existing benchmarks 5 on both the 50% agreement data (88%) and the combined data at all agreement levels (96%).

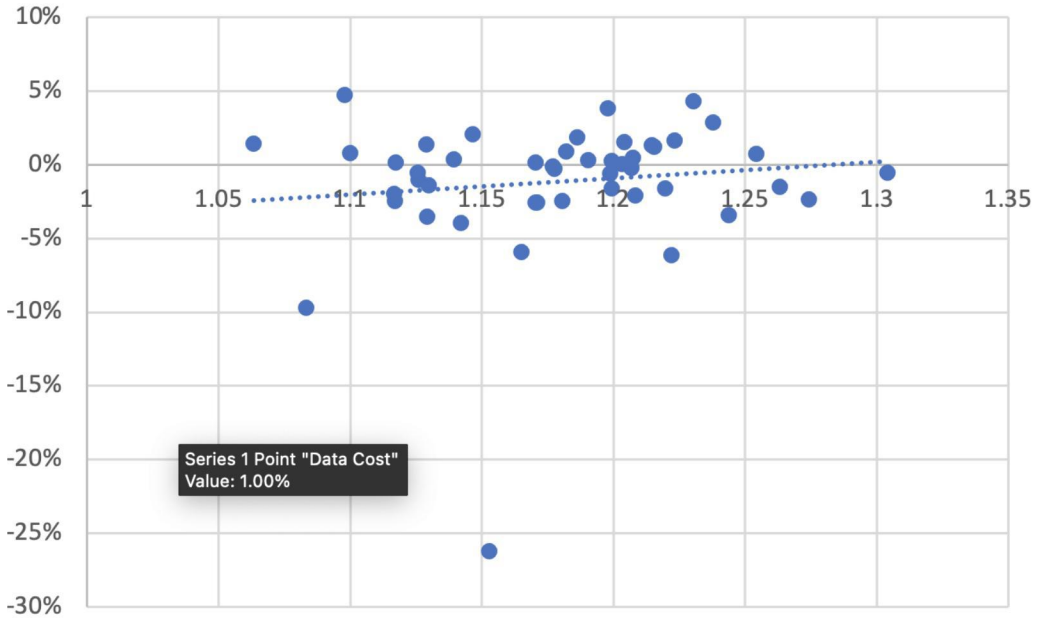

Figure 6: Predicted earnings transcript sentiment score vs. average position change for 50 randomly selected S&P tickers.

Furthermore, our average sentiment earnings transcripts shows a slight positive correlation with position changes (in number of shares) across the universe of US asset managers 6

6 Analysis

We believe that the use of a distilled model contributes significantly to our superior performance. In fact, it allows the model to focus on classification and lets us train for significantly more epochs. Each training epoch on the full dataset takes roughly 3 minutes to run.

While the predictive power of earnings call sentiment scores on position changes is rather disappointing (0.15 correlation), it makes sense overall. First, while it is established that a large fraction of a stock's under/over-performance occurs around earnings, it is not necessarily the case for position changes. Second, earnings transcripts are publicly available and hence it makes sense that their predictive power is limited. Last, the model was trained on sentiment prediction rather than position forecasting and we might have reached better results had we trained on position forecasting directly. In this vain, we note that scores range from 1 to 1.3, which is narrow. We believe this is due to the large amount of noise in transcripts (e.g. names, greetings, introductions) that is amplified by averaging sentiment as opposed to weighing certain sentences more than others.

7 Conclusion

In addition to improving performance on the Financial PhraseBank dataset to 96%, we believe that our project makes two original contributions in its approach. First, it tests the value of sentiment scores on the downstream task of predicting holdings. Second, instead of focusing on predicting stock market returns, we focus on predicting shares held by institutional investors, which we believe to be a better benchmark and objective for NLP models applied to equity markets.

8 Future work

There are several directions which we believe would be interesting to explore in the future, including:

- Testing a different model approach where instead of classification output, we use final layer hidden states from a text summarization model
- Experimenting with more informative, private datasets (e.g. self-sourced interview transcripts).
- Experiment with a meta-learning approach where each fund is treated as a separate meta-task. We believe this could help the model capture differences in style between asset managers. It also avoids comparing model output against stale, aggregated data.

References

[1] Robert P. Schumaker. An analysis of verbs in financial news articles and their impact on stock price. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Linguistics in a World of Social Media*, WSA '10, page 3–4, USA, 2010. Association for Computational Linguistics.

[2]

[3] Dogu Araci. Finbert: Financial sentiment analysis with pre-trained language models, 2019.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[5] Hugging Face. Transformers. `https://github.com/huggingface/transformers`.

[6] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, 2020.

[7] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015.

[8] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics.

[9] Matthew Backus, Christopher T Conlon, and Michael Sinkinson. Common Ownership Data: Scraped SEC form 13F filings for 1999-2017, 2020.

[10] Pekka Malo, Ankur Sinha, Pyry Takala, Pekka Korhonen, and Jyrki Wallenius. Good debt or bad debt: Detecting semantic orientations in economic texts. *Journal of the American Society for Information Science and Technology*, 04 2014.

[11] Financial Modeling Prep. `https://financialmodelingprep.com/developer/docs/`.

[12] `https://www.sec.gov/edgar/searchedgar/companysearch.html`.

[13] `https://pypi.org/project/sec-edgar-downloader/`.