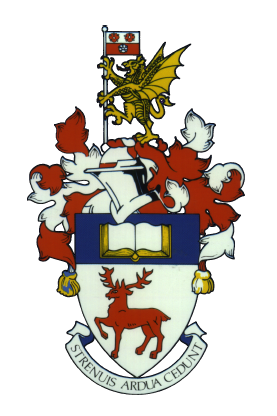
University of Southampton

Faculty of Physical and Applied Sciences

Electronics and Computer Science

**FinDeBERTa: Deep Bidirectional Transformers for Financial Sentiment Analysis**



Author: Roshan Patel

Project Supervisor: Adam Prugel-Bennett

Second Examiner: Geoff Merrett

December 06, 2020

A project report submitted for the award of BSc (Hons) Computer Science

# Abstract

The paper Attention is all you need (2017) [1] utilised an attention mechanism called the transformer to learn contextual relations between words in natural language processing tasks. The result was BERT, a model whose transformer reads entire sequences of words at once, producing benchmark BLEU scores on common translation tasks with minimal training. Despite this, BERT remains an underexplored terrain in the field of stock trend prediction.

Sentiment analysis is a challenging problem in stock trend prediction, and the bidirectional nature of BERT may offer improvements to gauging market sentiment compared to models that do not utilise transformers.

The paper processes a tailored dataset comprised of various sources including news headlines, social media and search engines with different flavours of BERT [2], [3]. The results will be evaluated, and the best model chosen, then refined using statistical market indicators. Models may be improved by establishing new target vectors. The final model’s performance is to be evaluated by back testing over a fixed period using the model as a trading strategy.

# List of Abbreviations

**BERT** = Bidirectional Encoder Representations from Transformers

**NLP** = Natural Language Processing

**DJIA** = Dow Jones Industrial Average

**ML** = Machine Learning

**MSE** = Mean Squared Error

**SGD** = Stochastic Gradient Descent

**SOTA** = State of The Art

**NYSE** = New York Stock Exchange

**NASDAQ** = National Association of Securities Dealers Automated Quotations Exchange

# Chapter 1

# Introduction

## **1.1 Motivation**

Predicting the daily directional changes of a stock price is a problem that many traders face. Aside from fundamental and technical indicators, market sentiment plays a significant role in the price of a stock. Being able to efficiently gauge market sentiment by considering information from a variety of useful sources and applying effective natural language processing techniques to accurately assess sentiment is a challenging problem. In the breakthrough paper *Attention is all you need* (2017) [1], researchers used Bidirectional Encoder Representations from Transformers (BERT) to set new benchmarks for common NLP translation tasks, spending comparatively low training time to achieve favourable performance. Despite this, BERT remains a relatively underexplored terrain in the field of stock trend prediction.

## **1.2 Objectives**

The project will utilise BERT to attempt to form a model that is able to capture market sentiment and will then include other technical indicators to develop a more complete model. The objectives set out for this project are as follows:

1. Data Gathering: Market sentiment data will be web scraped from 3 different sources: social media, news headlines and search engine results. The current aim is to have two types of each source and run scripts daily at 830am EST, capturing 50 results per source. This will be performed continuously over two months resulting in a dataset of 18,000 entries. This is heavily subject to modification and will be influenced by continued research and model performance.
2. Sentiment Analysis: Collected data will be cleaned and pre-processed, after which BERT will be used on Google Cloud Platform (GCP) Compute Engine to produce a vector score for each of the dated string entries. Preliminary tests on a contingency Kaggle data set of similar size took 1.5 hours using a single GPU Tesla P100 16GB.
3. Machine Learning: A model using the vectorised output from BERT will be trained to predict the direction and magnitude of the chosen stock/s closing price for the current day of trading. At this point different flavours of BERT, model tweaking, or additional training may be implemented to try and achieve a favourable performance. Technical indicators such as *simple moving average* will then be included in a new iteration of the model.
4. Comparison and Evaluation: The overall performance of the model will be evaluated. At this point in time three potential metrics have been identified to track overall model performance.
   * 1. The percentage of correct predictions over a fixed N number of days
     2. The Mean Square Error (MSE) of the model prediction compared with the actual price.
     3. The Backtrader Python library which facilitates implementation of a trading strategy to evaluate performance.

# Chapter 2

# Background Theory

## **2.1 The Stock Market**

The stock market is a publicly accessible exchange where buyers and sellers can purchase securities[[1]](#footnote-1), most commonly in the form of common shares, which represent part ownership of a company. The trading management of these publicly listed companies is conducted by an exchange, the two largest of which are the NYSE and the NASDAQ. To be traded publicly, a company must become listed, which is a lengthy process comprising of an initial public offering (IPO). In an IPO the private company meets with an investment bank and makes certain decisions, such as the number of shares, share price based on factors such as the debt to equity ratio, revenue consistency and many more [4]. From here the investment bank takes on the task of underwriting, assuming legal responsibility for the shares and dictating a set of terms required for listing, which will include the exchange’s own terms such as minimum number of shareholders to qualify for listing. There are many benefits to becoming publicly listed, including increased prestige and diversified ownership, however, there are also many shortfalls such as forced public disclosure, pressure for short term growth and potentially making decisions with greater consideration of market sentiment, favouring market price over intrinsic value.

## **2.1.1 Forces That Move Common Stock**

The price at any given moment is a result of the supply and demand at that point in the market. However, there are 3 main factors that influence the stock price of a

company: *Fundamental Factors*, *Technical Factors*, and *Market Sentiment*. *Fundamental factors* refer to a combination of two things, which have a variety of metrics to determine them:

1.) The earnings base of the company.

2.) The valuation multiple (expectations about the future).

*Technical factors* are external conditions that affect the supply and demand of a company’s stock price, such as inflation [4]*.* Finally, *market sentiment* refers to the general outlook of investors toward a particular company and can be influenced by a variety of fundamental and technical factors. With the rapid availability of news as well as opinions on forums and social media constantly being shared and considered, market sentiment plays a large role in the volatility of a stock [5].

## **2.1.2 Moving Average Convergence Divergence (MACD)**

The MACD is a momentum indicator that highlights the relationship between two moving averages. It is calculated by subtracting the 26 day Exponential Moving Average (EMA) from the 12 day EMA.

The Exponential Moving Average shows how the price of a security changes over a certain period of time. It is primarily chosen over other forms of moving average due to its property of reacting more to recent price changes. It is calculated as follows:

1. Obtain the first EMA values by taking the Simple Moving Average (SMA).
2. Calculate a smoothing constant defined as

The MACD is then calculated as follows:

From here, a signal line is plotted which is the 9 day EMA of the MACD. The MACD and Signal plots can then be used to provide buy and sell signals for traders. When the MACD value crosses above the signal line this signals the security is bullish[[2]](#footnote-2) and when the MACD crosses below the signal line it indicates the security is bearish[[3]](#footnote-3).

## **2.1.3 Relative Strength Index (RSI)**

The RSI is a momentum indicator primarily used for the evaluation of overbought or oversold securities. It is measured between 0 and 100, where common interpretation is that a value below 30 indicates an oversold condition and over 70 indicates that a stock is overbought. A trader will usually want to purchase a stock when it is oversold, as it suggests the stock is undervalued, and conversely may want to avoid a stock that is overbought, as it may soon experience a corrective pullback in price.

The RSI is calculated by first calculating the Relative Strength (RS):

The difference in daily closing price changes are determined and a Simple Moving Average of these changes are taken over a time period, most commonly 14 days. From here, the average gain is determined, which is equivalent to the mean of the upward changes. Likewise, the average loss is determined, which his equivalent to the mean of the downward changes. The Relative Strength is then normalised to a value between 0 and 100 to determine the Relative Strength Index as follows:

## **2.2 Artificial Intelligence (AI)**

The ability of machines to perform Artificial Intelligence can be broken down into 3 categories:

1. Artificial Narrow Intelligence – machines that operate within a pre-defined, pre-determined range, even if it has the appearance of being much more sophisticated.
2. Artificial General Intelligence – machines that exhibit human intelligence, successfully performing any intellectual task that a human being can.
3. Artificial Super Intelligence - *“any intellect that greatly exceeds the cognitive performance of humans in virtually all domains of interest” – Prof. Nick Bostrom, University of Oxford .*

Currently humans have created machines that can perform at the Narrow level of AI. Artificial Intelligence can be partially broken down into the following sub-fields that are pertinent to the content discussed in this paper:

Diagram

Description automatically generated

## They are defined as follows:

* Machine Learning – algorithms that improve through experience and by the use of data.
* Supervised Learning – algorithms match an input to an output, where the true output is labelled in the training data allowing the algorithm to evaluate itself on the training data.
* Unsupervised Learning – algorithms match an input to an output, where the true value for the output is unknown and so the algorithm tries to extract patterns on its own.
* Deep Learning – inspired by the biological structure of the human brain to make use of artificial neural networks.
* Natural Language Processing – branch of artificial intelligence that aims to help computers understand, interpret and manipulate human language.
* Sequence Classification – the task of dividing sequences of inputs into predefined categories.

All of these elements of artificial intelligence contribute to the ternary sequence classification task of correctly classifying financial text as either negative, neutral or positive.

## **2.2 Long Short-Term Memory (LSTM)**

The LSTM is a form of Recurrent Neural Network (RNN). RNN’s were developed to resolve the issue of Artificial Neural Networks not being able to leverage previous input values to inform later decisions. This was particularly damaging for time-series tasks such a stock price forecasting, where techniques such as trend analysis and future prediction rely on some form of memory mechanism for past data. RNN’s. achieved this by introducing loops.



Though the RNN allowed connecting previous information to the present task, it still suffered when it came to using information from states that were too far back due to vanishing gradients and exploding gradients. LSTMs were explicitly created to avoid this long-term dependency problem. This was achieved by adapting the RNN’s repeating module from a single tanh neural network layer to a four layered neural network.



There are three layered gates that control the cell state. These gates are: the forget () gate, the input gate () and the output gate (). The forget gate is a sigmoid layer that decides whether a piece of information that has been stored is still relevant. The input gate is also a sigmoid layer that decides what new information is significant enough to be stored in the cell state. The resultant state will be passed to the output gate and filtered with a sigmoid layer to decide the output contents via a function setting the values to either -1 or 1 and dictating whether the information is output or not.

Where: Neuron gate weight

Previous Cell Output

Current Input

Gate Biases

Neuron gate weight, Gate biases, Sigmoid activation function

## **2.3 Sequence Classification with Pre-Trained Language Models**

This section will provide a background of the neural architectures implemented in this paper and aim to provide a narrative of the progression of NLP models in the last few years.

## **2.3.1 Global Vectors for Word Representation (GLoVE)**

In order for words to be interpreted by machine learning models, they need a form of numeric representation for models to use in calculations. This was achieved by using a vector to represent words which simultaneously enables capturing semantic relationships between them, such as the semantic relationship between the words ‘London’ and ‘England’ or the syntactic equivalence between ‘had’ / ‘has’ and ‘was’ / ‘is’.

Researchers realised using embeddings pretrained on incredibly large datasets was an effective way to capture these relationships, which inspired GLoVE. GLoVE is an unsupervised learning algorithm for generating vector word representations trained using an LSTM. The algorithm uses the frequency and relative position of the words in a large corpus to measure the semantic similarity between corresponding words. This is achieved by using a corpus to form a global word-word cooccurrence matrix. The matrix is factorised to yield a lower-dimensional matrix by normalising the counts and log-smoothing them. The word embeddings are context independent and as such, there is just one word vector representation for each word and the model is not robust to polysemy[[4]](#footnote-4). The rationale is that the probability ratios have encoded meaning, demonstrated by the example below recreated from the original paper:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Probability and Ratio | k = solid | k = gas | k = water | k = fashion |
| P(k|ice) |  |  |  |  |
| P(k|steam) |  |  |  |  |
| P(k|ice)/P(k|steam) |  |  |  |  |

In the figure above the values plotted indicate the probability the next word is either ‘solid’, ‘gas’, ‘water’ or ‘fashion’. In the final row the ratio of P(k|ice) divided by P(k|steam) is shown. Observing the results it is evident that when the next word is closer to the numerator (ice) the ratio is >1 and when the next word is closer to the denominator (steam) the ratio is <1. The probability increases as the ratio gets further from 1. For completely dissimilar words, the value will be close to 1 as both probabilities should both be very low, evidenced by the ratio for ‘fashion’.

## **2.3.2 Embeddings from Language Models (ELMo)**

ELMo builds upon GLoVE by using a contextualised embeddings instead of a fixed embedding for each word. This is achieved by using a bi-directional LSTM to analyse an entire sequence of words before assigning each one an embedding. This contextual understanding is gained by training ELMo on a task called language modelling, whereby the objective is to predict the next word in a sequence of words. The implementation of the bi-directional LSTM was integral to the success of ELMo as it enabled the model to be able to observe the previous and next words as well as train by analysing the sequence from **both directions**. Once the forward and backward language models are trained, ELMo concatenates the hidden layer weights together into a single embedding.

## **2.3.3 Universal Language Model Fine-Tuning (ULMFiT)**

The goal of ULMFiT was to build one model that could serve any classification task. It was primarily achieved using transfer learning[[5]](#footnote-5) and is regarded as one of the first successful implementations of transfer learning in natural language processing. The ULMFiT paper highlighted two novel techniques that improved the transfer learning process: Slanted Triangular Learning Rates and Discriminative Fine-tuning. These are leveraged in the implementation section where there are explained in more detail. As the name suggests ULMFiT is initially pre-trained on a **language model**ling task utilising a large Wikipedia dataset, and adaptation to new tasks can be achieved by further fine-tuning.

## **2.3.4 Bidirectional Encoder Representations from Transformers (BERT)**

In 2017, Google Research published a paper titled Attention Is All You Need in which they proposed a new architecture known as the Transformer for the purpose of Neural Machine Translation (NMT). The Transformer took inspiration from the observation that the best performing models at the time utilised an encoder and a decoder commonly connected through a mechanism known as attention. Transformers capitalise on this and implement a connected encoder and decoder while dispensing of recurrence entirely. The result led to a paradigm shift in natural language processing and the vast majority of SOTA performances on NLP tasks today are from transformer based models.

Unlike LSTMs, Transformers do not need sequential data to be processed in order, as the entire sequence is encoded at once. Additionally, utilisation of attention mechanisms addresses the previously discussed problem of vanishing and exploding gradients in RNN architectures, as unlike RNN architectures, the attention layer can access all states and weight them using a learned relevancy of all the states with respect to the current token being observed. A self-attention mechanism enables identification of context for any position in the input sequence, allowing Transformers more training parallelisation compared to LSTMs. This drastically reduces training time, which enables training on larger datasets than before.

The transformer is composed of an encoder and a decoder:

The BERT model utilises the encoder by stacking multiple layers of them. Each encoder can be broken down into the feed forward and self-attention sub-layers. The input is a combination of token embeddings, segment embeddings and positional embeddings. The self-attention layer is where the encoder observes surrounding words in an input sequence to generate a set of **contextually aware** output encodings. The outputs are fed into the feed forward neural network which further processes the encodings.

In an attention unit, three weight matrices are learned: query weights, key weights, value weights. These weights are multiplied by each word embedding to produce vectors for the query, key and value. The self-attention calculation can be expressed as one matrix operation:

Where: The query vectors

The transpose of the key vectors

The value vectors

Dimension of the key vectors (input sequence length)

The transformer frames the encoded input as a set of key-value pairs , both of dimension . In the decoder, the previous output is compressed into a query.

The output of the encoder is fed into a classification layer where during pretraining, the model is trained on two separate tasks: Masked Language Model (MLM) and Next Sentence Prediction (NSP).

A picture containing diagram

Description automatically generated

The **Masked Language Model** task involves replacing 15% of the words at the input stage with a [MASK] token. The model then uses the context of the words available to it to predict the masked words. The **Next Sentence Prediction** task involves the model receiving two sentences and the task is to predict whether or not the second sentence follows the first sentence in the original input text. The goal of the pretraining stage of BERT is to minimise the combined loss function of both Mask Language Model and Next Sentence Prediction tasks.

## **2.4 Sentiment Analysis in the Financial Domain**

Sentiment analysis is the mining of text to extract subjective information, usually to create meaningful insights. Financial sentiment is predominantly done to use as a numeric representation in a strategy with the hope that that strategy will yield a positive Alpha[[6]](#footnote-6) ().

There are two main challenges with sentiment analysis in the financial domain: the first is correctly mining the intended sentiment from the text and the second is having a representative enough dataset so that the sentiment being captured is as representative of the problem as possible [6]

# Chapter 3

# Implementation

This section will detail the creation of the financial sentiment analyser FinDeBERTa and subsequently its utilisation in a trading strategy.

## **3.1 Proposed Methodology**

**Flow chart with machine learning model**

## **3.2 FinDeBERTa**

This subsection details the full implementation of the FinDeBERTa model including the approach to additional training for downstream classification tasks in the financial domain. It also covers the Bayesian hyperparameter tuning used for training optimisation and the empirical findings within the context of current literature. Consequently, this subsection aims to answer the following **research questions**:

1. What is FinDeBERTa’s performance on sequence classification tasks in the financial domain when compared to other transfer learning methods?
2. How does FinDeBERTa compare to SOTA methods in multiclass financial sentiment sequence classification?
3. What is the performance improvement of FinDeBERTa after Bayesian hyperparameter optimisation?

## **3.2.1 The Financial Phrase-bank**

The Financial Phrase-bank **Source** is an annotated corpus of 4840 labelled financial phrases. The phrases are from English news articles concerning listed companies in OMX Helsinki. This dataset was chosen partly due the text sequences being semantically similar to the target news articles that will be analysed by the transformer model. Moreover, this dataset was selected for its high reliability, as sixteen members comprised of researchers and postgraduate students on financial programs were responsible for labelling the data. The dataset is split into four different groups differentiated by the % agreement on the sequence label: *negative*, *neutral* or *positive*. The dataset is divided into an 80-10-10 train-test-validation split, and all experimental analysis is conducted with 10-fold cross validation.

|  |  |
| --- | --- |
| Dataset | Number of Sentences (% Negative, % Neutral, % Positive) |
| 100% agreement | 2259 (25.2, 61.4, 13.4) |
| >75% agreement | 3448 (25.7, 62.1, 12.2) |
| >66% agreement | 4211 (27.7, 60.1, 12.2) |
| >50% agreement | 4840 (28.2, 59.3, 12.5) |

## **3.2.2 Evaluation Metrics**

The evaluation metrics used are: Accuracy, Macro-averaged F1 score, Weighted F1 score: **EXPLAIN FP FN etc and do confusion matrix thing**

**Accuracy:**

**Macro-averaged F1:**

The macro-averaged f1 score involves calculating the f1 score across each class: negative, neutral and positive, individually. The scores are then summed and divided by 3. Through this, an unweighted (macro) f1 score is produced whereby each class is considered equal in size, meaning class imbalance is not accounted for. Although the Financial Phrase-bank dataset is imbalanced, this is likely the most useful f1 metric, as the class distribution of negative, neutral and positive sentiment in forums and news articles will vary depending on current events at the time.

**Weighted F1:**

The weighted f1 score is calculated similarly to the macro-averaged f1 score. The difference being instead of dividing the summed f1 score by the number of classes, each score is multiplied by the class’ ratio in the dataset. The class ratios for the full dataset of >50% agreement have been inserted into the equation above to illustrate this; 28.2% *negative*, 59.3% *neutral*, 12.5% *positive*.

## **3.2.3 Baseline Methods**

In order to adequately assess FinDeBERTa’s performance on the Financial Phrase-bank dataset, baselines are gathered from four other implementations:

**Bi-LSTM with GLoVE.**

For this implementation, a bi-directional LSTM classifier is implemented with a hidden size of 64 and hidden state size of 128. The last hidden state is mapped to a vector with 3 values by a fully connected feed-forward layer. These 3 values represent the probability of each of the 3 class labels. A dropout layer is included with a dropout probability of 0.3. The optimiser used is the Adam optimiser. The learning rate is set to 3e-5 with a warmup ratio of 0.1. The model was run for 50 epochs with early stopping exercised when there was no improvement on the validation loss for 3 consecutive iterations. The embeddings used for this implementation are the GLoVe embeddings.

**Bi-LSTM with ELMo.**

This implementation is almost identical to the one above, the only difference being ELMo embeddings are used instead of GLoVe embeddings. This is done to observe any advantage of contextualised word representations, as with ELMo the surrounding words influence the representation of the word.

**ULMFit.**

The pretrained ULMFit language model is fine-tuned for classification with the Financial Phrase-bank dataset using the *Fast.ai* library. A fully connected layer is added to the output of ULMFit. Training is conducted for 15 epochs with a learning rate of 4e-3 and an Adam optimiser. The early stopping requirement is the same as the previous two implementations.

**BERT-Base-Cased.**

The bert-base-cased model is a 120-layer, 768-hidden, 12 head, 109M parameter model pretrained on cased English text. It is implemented using the *Huggingface* library and fine-tuned over 2 epochs with a learning rate of 2e-5 and warmup ratio of 0.06 as recommended by Google Research. The Adam optimiser is also used for this implementation.

## **3.2.4 Implementing FinDeBERTa**

Firstly, publicly available SOTA models were assessed based on two factors:

1. Their performance on the GLUE MNLI benchmark task.
2. Their performance by validation loss after further training on the Financial Phrase-bank corpus.

The General Language Understanding Evaluation (GLUE) benchmark is a collection of resources for training, evaluating and analysing natural language understanding systems. Performance on the Multi-Genre Natural Language Inference (MNLI) benchmark was chosen as an evaluation metric as the task is similar to the Financial Phrase-bank task. The similarity is that both objectives are to correctly sort the text sequences into 3 distinct classes. Specifically for MNLI, “given a premise sentence and a hypothesis sentence, the task is to predict whether the premise entails the hypothesis (entailment), contradicts the hypothesis (contradiction), or neither (neutral)”.

After reviewing the scores of various models, 3 were selected and further trained on the Financial Phrase-bank corpus. The results are as follows:

|  |  |  |
| --- | --- | --- |
| Base Model | MNLI Score (acc) | FPB Val Loss (cross-entropy) |
| RoBERTa-Base/Large | 87.6/90.2 | 0.408/0.389 |
| ELECTRA-Base/Large | 88.8/90.9 | **0.400**/0.381 |
| DeBERTa-Base/Large | **88.8**/**91.3** | 0.406/**0.377** |

These preliminary results show that DeBERTa is likely to perform marginally better than the other models and as such it was selected as a base architecture.

Due to the aforementioned similarities between the MNLI task and the Financial Phrase-bank task, it was hypothesised that additional training on the MNLI set could be beneficial to the overall performance of FinDeBERTa. Specifically, the large corpus size of 393,000 train and 20,000 test samples coupled with the high data reliability implied by its use as a classification benchmark, suggested that further training on the dataset could improve the model’s generalisation to ternary sequence classification problems. The model submitted by Microsoft for assessment on the MNLI test set was accessed and implemented through the *Huggingface* library. The results after further training on the Financial Phrase-bank dataset are as follows:

|  |  |
| --- | --- |
| Base Model | FPB Val Loss (cross-entropy) |
| DeBERTa-MNLI-Base/Large | 0.361/0.344 |

**Interpret results**

## **3.2.5 Hyperparameter tuning with Bayesian Optimisation**

The objective function of the optimisation problem for this model is a black box function. As such, there is no analytical expression for and its derivatives are unknown. Evaluation of the function is conducted by sampling various hyperparameter values . Bayesian optimisation was deemed to be the most useful method of hyperparameter tuning here. This is due to its incorporation of the *acquisition function* to propose promising sampling points in the search space, thereby markedly reducing search time compared to Grid and Random search.

Chart

Description automatically generatedThe search was run using Southampton University’s IRIDIS 5 supercomputer cluster, where 4 independent agents were deployed to run Bayesian searches simultaneously. Initially, a Bayesian search was done with the learning rate and number of epochs to narrow down the subsequent search space, as these parameters are of primary importance to model performance **SOURCE**. A random sample of the results used to inform the subsequent search are shown below:

The eval\_loss label refers to the cross entropy loss on the validation dataset. This initial search indicated that generally the best results had a learning rate between 1e-6 and 3e-5 and ran for 1 to 3 epochs. These parameters were restricted in the subsequent search. The parameters and their ranges in the subsequent search are outlined below.



**Explain parameters show results.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | All Data | | | 100% Agreement | | |
| Model | Loss | Accuracy | Macro-F1 | Loss | Accuracy | Macro-F1 |
| Bi-LSTM with GLoVE | 0.80 | 0.70 | 0.64 | 0.57 | 0.81 | 0.74 |
| Bi-LSTM with ELMo | 0.69 | 0.76 | 0.71 | 0.50 | 0.84 | 0.77 |
| ULMFit | 0.44 | 0.81 | 0.78 | 0.20 | 0.93 | 0.91 |
| BERT | 0.41 | 0.83 | 0.81 | 0.20 | 0.94 | 0.92 |
| LPS | - | 0.71 | 0.71 | - | 0.79 | 0.80 |
| HSC | - | 0.71 | 0.76 | - | 0.83 | 0.86 |
| FinSSLX | - | - | - | - | 0.91 | 0.88 |
| FinBERT (Prosus AI) | 0.37 | 0.86 | 0.84 | 0.13 | 0.97 | 0.95 |
| FinDeBERTa-Base |  |  |  |  |  |  |
| FinDeBERTa-Large |  |  |  |  |  |  |

**Evaluation (Address 3 research questions)**

## **3.3 Stock Prediction Model**

This subsection details the creation of the stock prediction model beginning with the data collection step. Next, sentiment analysis is conducted using FinDeBERTa and statistical analysis is conducted using the **METRICS** metrics. The resultant features are used in a machine learning model.

## **3.3.1 Data Collection**

The data collection process was conducted with the objective of collecting as much speculative data about general market sentiment and NASDAQ:FB as possible over the 750 day period. In an attempt to capture a fully representative dataset containing both entity based and crowd based perception **Source**, data was sourced from a combination of news providers as well as the discussion forum Reddit.

A total of 57 news providers were web scraped for articles. This included both well-established fact reporting entities [ ‘CNBC’ , ‘Reuters’, ‘Forbes’ ] as well as more speculative entities [ ‘Seeking Alpha’, ‘Market Watch’, ‘InvestorPlace’ ]. This was done with the intention that sentiment scores could be separated by source and treated as individual or aggregated hyperparameters. Additionally, general news and company specific news was captured independently in order to capture the general market sentiment as well as the specific sentiment for Facebook. Python scripts were executed once per day at 0925 EST. A full list of news providers is available in the Appendices.

With regards to the Reddit data collection, five subforums or ‘subreddits’ were chosen to collect data from. The five subreddits were chosen based on a selection criteria that considers the size of the subreddit community as well as the mean number of monthly posts returned by the Regular Expression **Put search regex here.** The subreddits selected were: /r/investing, /r/wallstreetbets, /r/stocks, /r/options and /r/CryptoCurrency.At the time of collection the subreddits had a combined number of 17.4 million members.

The full dataset is comprised of 103,334 Reddit comments, 7,749 Facebook specific articles and 28,962 general market articles.

1. Security – A tradeable financial asset, usually in the form of a stock, bond, or option [↑](#footnote-ref-1)
2. Bullish – characterised by rising share prices [↑](#footnote-ref-2)
3. Bearish – characterised by falling share prices [↑](#footnote-ref-3)
4. Polysemy – the coexistence of many possible meanings for a word or phrase [↑](#footnote-ref-4)
5. Transfer learning – A machine learning method where a model developed for a task is reused as the starting point for another task [↑](#footnote-ref-5)
6. Alpha – The performance of investment against a market index or benchmark return [↑](#footnote-ref-6)