Machine Learning in Sentiment Analysis

How does the performance of a Bidirectional LSTM neural network compare to that of a LinearSVC in terms of classification accuracy, precision, recall, and F1 Score when analysing sentiments in financial news?

Vu Nguyen Le

0 Table of contents

0. **Table of contents** **2-3**

1. Introduction **4-5**

1.1. Societal application and importance of sentiment analysis 4

1.2. Rationale and Aim 5

2. Theoretical background **6-12**

2.1. Artificial Neural Networks

Layers and Neurones 6

Types of Activation function and Sigmoid layer 8

Learning, Loss function, and Training 9

2.2. Recurrent Neural Networks and LSTM 10

2.3. Support Vector Machines 10-11

2.4. Literature Review and Hypothesis 11-12

3. Theoretical background **12-19**

3.1. Software and Program 12

3.2. Coding language and Libraries 13

3.3. Dataset 13

3.4. Procedure 14-15

3.5. Assumptions 16

3.6. Variables .....

Independent Variable 16

Dependent Variable 16-17

Control Variable 17-19

4. Experimental Data **20-24**

4.1. Raw Data 20-21

Vu Nguyen Le

4.2. Processed Data 21-24

5. Conclusion 24

6. Evaluation 24-26

7. References 27-28

8. Bibliography 29-30

9. Appendix 31-38

Vu Nguyen Le

1 Introduction

1.1 Societal application and importance of sentiment analysis

Analysis of sentiments, also known as opinion mining, allows the understanding of human emotions and opinions in textual data. In recent years, sentiment analysis has achieved widespread applications, from customer feedback analysis and product reviews to political media tracking and, most interestingly, financial market analysis.

Understanding sentiments in financial news articles is crucial for investors, traders, and other relevant stakeholders to make informed trading decisions, such as predicting the trend of the equity market, analysing the earning reports of firms, and creating an automated stock trading system. Rumours, public opinions, and breaking news often influence these articles. Sentiment analysis offers a mechanism to quantify and interpret the impacts of these influences holistically on the overall message. An application of sentiment analysis is bias detection algorithms, which are often used by breaking news media, such as Ground News – one of my favourite breaking news services. This is because Ground News employ biased ratings from analysing sentiments to assess political articles and classify them on a spectrum from left-wing to right-wing.

The effectiveness of sentiment analysis depends on the Machine Learning (ML) model's ability to assess contextual data, which is a procedure of Natural Language Processing (NLP). A traditional ML model for analysing sentiments was that of a Support Vector Machine (SVM), which "distinguishes between two classes by finding the optimal hyperplane that maximizes the margin between the closest data points of

Vu Nguyen Le

opposite classes” (IBM, 2023). To this day, SVMs are still extensively used in fields like facial recognition, gene classification, and spam filtering systems. In contrast, modern Deep Learning models, such as Long Short-Term Memory (LSTM) networks and transformer models, can identify patterns based on the input data and make decisions with minimal human intervention (Team, 2022). Interestingly, LSTM network is known to “excel at capturing long-term dependencies, making it ideal for sequence prediction tasks" (Saxena, 2021). As such, LSTMs have been used as the system architecture behind famous ‘digital assistants’ such as Apple’s Siri and Amazon’s Alexa.

1.2 Rationale and Aim

Intrigued by the strengths and real-life applications of both models, this research aims to draw a comparative analysis of the performance of a traditional ML approach compared to that of a Deep Learning model, that being a LinearSVC implementation and a directional LSTM network for sentiment analysis on financial news.

LinearSVC was specifically chosen for its efficiency in handling high-dimensional feature spaces. Unlike other SVM variants, LinearSVC is optimised for linear classification problems and tends to be relatively faster than its non-linear classifier counterparts. On the other hand, a Bidirectional LSTM model - which processes input sequences in both forward and backward directions - would capture both the past and present context of the text (Saxena, 2021). Thus, a Bidirectional network can generate a more nuanced analysis than a unidirectional one.

Vu Nguyen Le

Therefore, this research explores the question: "**How does the performance of a Bidirectional LSTM neural network compare to that of a LinearSVC in terms of classification accuracy, precision, recall, and F1 Score when analysing sentiments in financial news?"**.

To address this, a comparative study is conducted using a public dataset of financial news headlines. The models are evaluated based on key performance metrics, including classification accuracy, precision, recall, and F1 Score, with their outputs tracked and visualised across multiple epochs. Through this, the research aims to determine which model is better suited for sentiment analysis in the financial domain, especially when dealing with the context-dependent nature of textual data.

2 Theoretical background

2.1 Artificial Neural Networks

A neural network is a category of ML that classifies or creates data based on a network of neurones that learns from a particular set of inputs. The fundamental architecture is an Artificial Neural Network (ANN), which consists of nodes connected through a multitude of layers. Recurrent Neural Networks (RNNs) are a class of ANNs designed to recognise patterns in data sequences, and LSTMs are a type of RNN architecture. Thus, fundamental knowledge of neural networks is needed to understand the execution of the LSTM network.

2.1.1 Layers and Neurones

Vu Nguyen Le

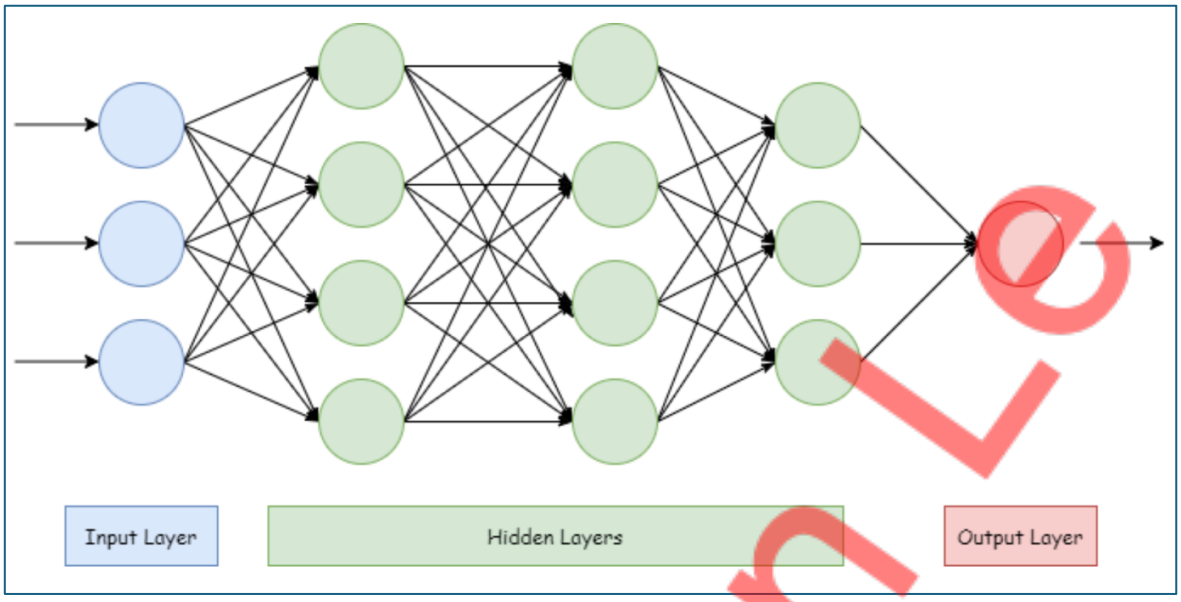


Figure 1. A model of the three layers that make up an ANN (Jakubiak, 2022).

Based on Figure 1, the neurones are arranged in a web-like structure where every node is connected to new nodes in the next layer. There are three categories of layers: an input layer, hidden layers (with varying numbers of layers), and an output layer. The input layer retrieves the data and passes it onto the first hidden layer. The hidden layers calculate output nodes based on the individual input nodes using an activation function. This activation function squeezes the unrestricted sizes of input neurones to output values only between 0 (false state) and 1 (true state). This also means that input neurones can take any arbitrary value so long as it belongs between 0 and 1. The output layer computes the values of the entire network and returns them as output to the data scientist.

Each neurone (node) consists of a weight and bias value, which are factors that decide the influences of inputs on the output value. Weights determine the strength of the connections between nodes, where the larger the weight, the bigger the impact that an input neurone of a previous layer has on the output neurone of the next layer

Vu Nguyen Le

through an activation function. This weight can be either positive or negative to strengthen or decrease the influence of the next node. Moreover, bias values entail how likely an input neurone is to output 1. At a given hidden layer, the processing of the neurone would be like:

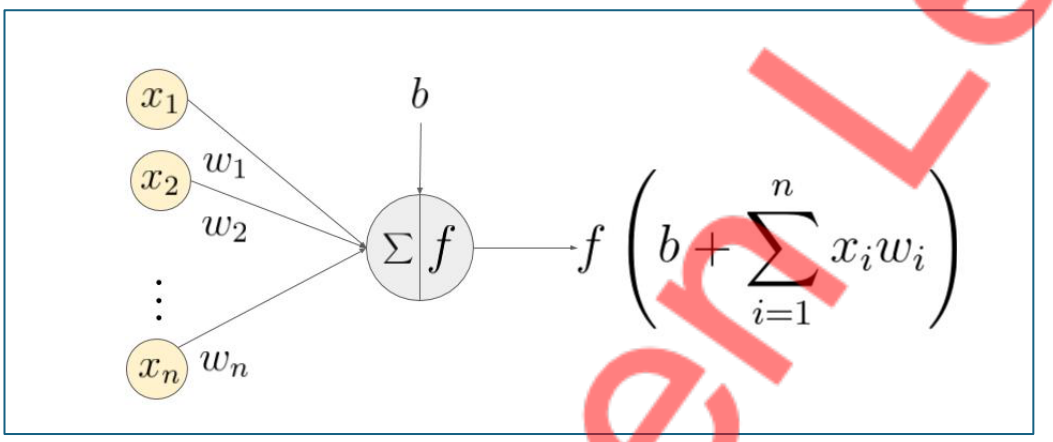


Figure 2. An abstract of an artificial neurone in the hidden layer (Learn OpenCV, 2017), with input nodes $x_1, x_2, x_3, \ldots, x_n$, weights $w_1, w_2, w_3, \ldots, w_n$, a bias b, and the activation function f applied to the weighted sum of the inputs.

Letting a variable z be the weighted sum $\sum_{i=1}^{n} x_i \cdot w_i$ then the output neurone can be expressed as an activation function $f(z)$. Here, the weights and the bias transform z linearly, known as a "feed-forward" control.

2.1.2 Types of Activation function and Sigmoid layer

RNNs and LSTMs use a non-linear activation function, such as the sigmoid function. The activated output of a sigmoid neurone $f(z) = \sigma(z)$, with inputs $x_1, x_2, x_3, \ldots, x_n$ weights $w_1, w_2, \ldots, w_n$, bias b, and indexed with j is:

$$\sigma(z) \equiv \frac{1}{1 + e^{-z}}$$

Vu Nguyen Le

$$\sigma(z) = \frac{1}{1 + \exp\left(-\sum_j w_j \cdot x_j - b\right)}$$

(Nielsen, n.d.)

If supposed a sigmoid neurone z is a large positive value, then $\sigma(z) \approx 1$, whilst if z is a large negative value, then $\sigma(z) \approx 0$. Here is the shape of the graph:

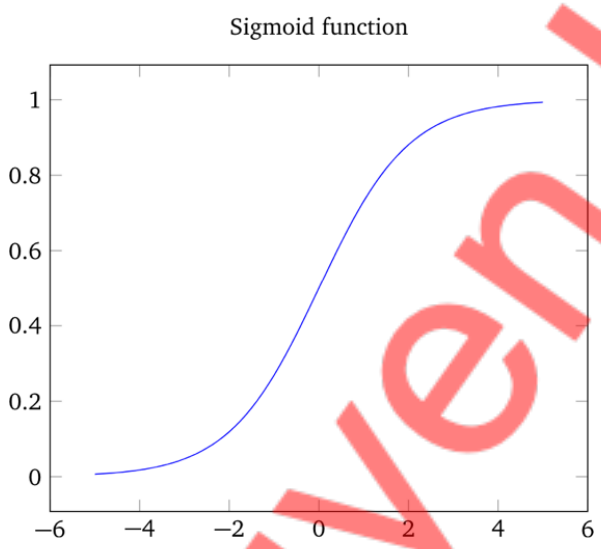


Figure 3. Diagram of the sigmoid activation function (Nielsen, n.d.), where the x-axis shows the domain of z values, whilst the y-axis shows the range of output $\sigma(z)$.

2.1.3 Learning, Loss, and Training

The process of network "learning" is by adjusting weights and biases throughout layers until the outputs are accurate for what is expected as a correct output (Jordan, 2017), which is also referred to as backpropagation. The network uses a loss function to calculate the 'loss' error (the difference between the predicted and desired output) from an iteration. The loss "error" is lower when the model makes fewer mistakes with its calculation. Hence, the training process is simply passing the generated loss through an optimiser algorithm to adjust the weights and biases that minimise the loss function over the training dataset.

Vu Nguyen Le

2.2 Recurrent Neural Networks and LSTM

RNNs handle sequential data by maintaining an internal state that can capture temporal dependencies between node points. The recurrent connections allow RNNs to retain information from previous time steps, allowing for backpropagation through time (BPTT, an extension of backpropagation). In other words, RNNs can capture long-term dependencies in sequential data but still suffer from vanishing gradient. LSTMs incorporate memory cells and gating mechanisms - specifically, the input, forget, and output gates - that regulate the flow of information. These gates allow LSTMs to selectively remember or forget information, effectively addressing the vanishing gradient problem. This makes LSTM a powerful model for tasks that require learning from long data sequences, such as sentiment analysis.

2.3 Support Vector Machines

SVM works by constructing a hyperplane that divides the feature space into two distinct regions that classify data points differently. The primary objective of SVM is to maximise the margin between the classes by finding the optimal hyperplane. The margin is the distance between the hyperplane and the nearest data points from each class, known as support vectors.

Linear Support Vector Classification (LinearSVC) is an implementation of SVM designed specifically for linearly separable data. Unlike traditional SVM, which can use complex kernel functions to handle non-linear data, LinearSVC uses only a linear kernel. This allows the LinearSVC to be computationally inexpensive compared to other SVM variants. Like neural networks, LinearSVC utilises a decision variable z

Vu Nguyen Le

linearly transformed by a weight w and a bias b. The objective function for LinearSVC that has a weight w and bias b as parameters can be expressed like the following:

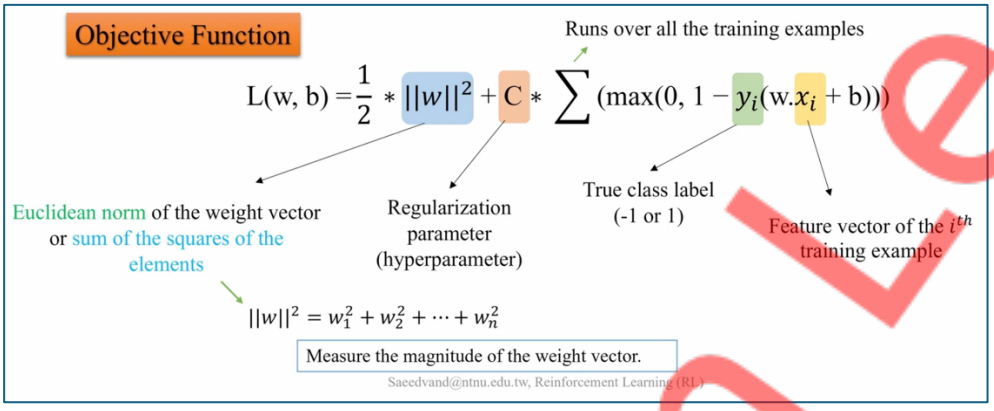


Figure 4. An image extract of the objective (loss) function and the representation of each variable from a lecture recording on SVM classification models (Saeedvand S., 2023).

To achieve the optimal hyperplane is to minimise the loss function $L(w, b)$ by reducing the hinge loss.

2.4 Literature Review and Hypothesis

The literature review on sentiment analysis reveals that LSTM networks were generally more accurate than SVM models. Firstly, a 2022 paper by Alfarizi et al. compared the performance of both models in classifying emotional text data. Their findings show that the LSTM model achieved an accuracy score of 97.50%, which outperforms the LinearSVC model's accuracy of 89%. Secondly, a 2020 paper by Lynch et al. reviewed various open-source ML algorithms for analysing sentiments on Twitter tweets (including both images and texts). They found that deep learning models, such as LSTM, that the LSTM-based model produced the "best results for positive and negative TSA", whilst SVC Perceptron algorithms were among those that

Vu Nguyen Le

produced favourable results for image sentiment analysis. Thirdly, a 2019 article by A. Zhang, which analysed sentiments on the presidential debate between Trump and Hilary found that the LinearSVC model achieved a mean accuracy of 63.76% whilst an LSTM model with Keras achieved an accuracy of 86.95% after 20 epochs.

Based on the existing literature, this research hypothesises that the Bidirectional LSTM model would outperform the SVM due to its capability to handle sequential dependencies, whereas the LinearSVC model may fall short due to their inability to model temporal dynamics.

3 Experimental Methodology

3.1 Software and Program

The program was coded using Jupyter Notebook Integrated Development Environment (IDE) on the Visual Studio Code (VSC) application. Jupyter Notebook's 'code cells' features, where each cell can be executed individually, and inline visualization allows easy troubleshooting and modification of the program. Moreover, VSC provides extensions and functionalities that make programming on it convenient and visually attractive. The entire program's code can be found in Appendix A. The program for pre-processing and implementation of LinearSVC was adapted from (shwethaspdevam, 2020) with the aid of Scikit-Learn 0.24.1 documentation for SVM. On the other hand, the program for the Bidirectional model was adapted from (Koushiki, 2024) with the aid of Keras 3 API documentation for Layers. Error debugging and further modifications were made with the feedback from ChatGPT-4o.

Vu Nguyen Le

3.2 Coding language and Libraries

Python v3.12 was the primary coding language for the program because of the simple syntax and abundance of libraries. Libraries were used to preprocess records, visualise data, and create the models. They include **NumPy** for calculations and numerical analysis; **Pandas** for loading dataset, cleaning, and transforming the data; **NLTK** for tokenising text, removing stop words, stemming, and lemmatisation; **Matplotlib**, **Seaborn** for exploratory data analysis (EDA); **Keras API** for building the Bidirectional LSTM network; **Scikit-Learn** for building the LinearSVC model.

3.3 Dataset

The dataset, which originally was “FinancialPhraseBank” and taken from the Kaggle website, holds 4846 records of financial news headlines. The dataset is representative of type of text that traders and investors rely on to gauge market sentiment. The sentiment labels are categorized into 3 types: positive, neutral, and negative. For simplicity, the data fields were renamed to “sentiment” and “headline” and the dataset was renamed to “sentimentDatabase.csv.”

	sentiment	headline
0	neutral	According to Gran , the company has no plans to move all production to Russia , although that is...
1	neutral	Technopolis plans to develop in stages an area of no less than 100,000 square meters in order to...
2	negative	The international electronic industry company Elcoteq has laid off tens of employees from its Ta...
3	positive	With the new production plant the company would increase its capacity to meet the expected incre...
4	positive	According to the company 's updated strategy for the years 2009-2012 , Basware targets a long-te...
5	positive	FINANCING OF ASPOCOMP 'S GROWTH Aspocomp is aggressively pursuing its growth strategy by increas...

Figure 5. Output from loading the first 6 records of the modified dataset.

3.4 Procedure

1. **Import** all the necessary Python libraries.

Vu Nguyen Le

2. **Load** the “sentimentDatabase.csv” file into a Pandas Data Frame.
3. **Visualise** the distribution of sentiment labels using Seaborn’s count plot.

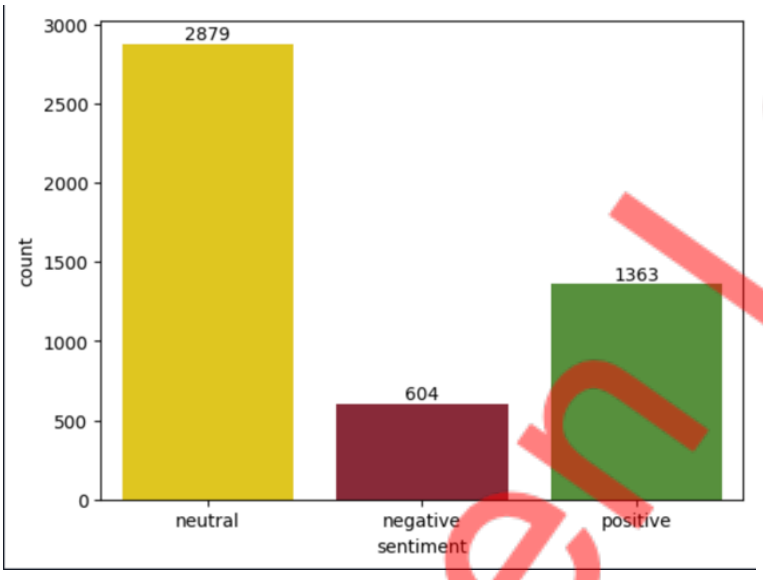


Figure 6. The visual count plot on the “sentimentDatabase.csv” dataset.

4. **Remove punctuation** marks using Regular Expression.
5. **Remove stop-words** such as "the", "is", and "these" using NLTK’s list of stop-words.
6. **Apply stemming** by using PorterStemmer to reduce words to their root form. For example, "running" becomes "run".
7. **Remove the 12 most frequently occurring words** that may not contribute to sentiment analysis.
8. **Convert words to their base form** (e.g., "better" becomes "good") using NLTK's WordNet Lemmatizer to standardize vocabulary.
9. **Convert categorical sentiment labels** into numeric a format using Scikit-learn's LabelEncoder.
10. **Split the preprocessed dataset** into training and testing sets with an 70-30 ratio.

Vu Nguyen Le

11. **Convert text data** into sequences of tokens using Keras' TextVectorization and pad the sequences to equal length.
12. **Build the Bidirectional LSTM model** with Keras, including an embedding layer, dropout for regularization, and dense layers.
13. **Train the LSTM model** for 10 epochs with early stopping mechanism to prevent overfitting.
14. **Apply TF-IDF vectorization** to convert the text data into numerical features for use with the LinearSVC model.
15. **Use a pipeline to vectorize the data** and fit the LinearSVC model for the same number of iterations (epochs) that the LSTM successfully trained with.
16. **Evaluate** the trained LinearSVC model and LSTM model on the test set, calculating classification accuracy, F1 Score, precision, recall *per epoch*.
17. **Plot all the relevant performance metrics** using Matplotlib and **visualize the Confusion Matrix heatmap** using ConfusionMatrixDisplay for both models.

3.5 Assumptions

It is assumed that the sentiment labels in the dataset accurately represent the true sentiments of the financial news headlines, as manually verifying over 4800 records is not feasible. However, the dataset is still likely to be credible as it has been extensively used in over 30 sentiment analysis projects on Kaggle.

Vu Nguyen Le

3.6 Variables

3.6.1 Independent Variable

The independent variable is the number of epochs, or, in other words, the number of complete iterations of the training dataset (Daniel, 2023). Epochs can be varied by adjusting the parameter of the models when programming with the Keras or Scikit-Learn modules. For the LinearSVC class in the Scikit-Learn library, the number of epochs is the parameter "max_inter". The number of epochs ran was 6 epochs. This moderate range was chosen to see the initial learning curve and the point at which the models start to converge.

3.6.2 Dependent Variable

A wide range of performance metrics were chosen for model evaluation including precision and recall, classification accuracy and F1 Score, error loss function (only for LSTM), and confusion matrix heatmap. Precision measures the proportion of positive identifications that were correct. Recall measures the proportion of actual positives that were correctly identified. The F1 Score is a harmonic mean between precision and recall. All calculations for these metrics depend on the following four prediction outcomes:

Prediction outcomes	Abbreviation	Model description
True Positive	TP	Prediction is **positive** Actual outcome is **positive**
True Negative	TN	Prediction is **negative**

Vu Nguyen Le

		Actual outcome is **negative**
False Positive	FP	Prediction is **positive** Actual outcome is **negative**
False Negative	FN	Prediction is **negative** Actual outcome is **positive**

Figure 7. The 4 cases of prediction outcomes. "Actual outcome" refers to the original sentiment of a given data record whilst "Prediction" is the model's output.

The confusion matrix is a display of the compositions of TP, TN, FP, and FN. Knowing these four prediction outcomes, the general expression for the other metrics can be expressed as:

$$Classification\ Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F_1 = \frac{TP}{TP + \frac{1}{2}(FP + FN)}$$

(Kundu, 2022)

3.6.3 Control Variable

Any variations in the performance metrics must be because of the number of epochs. To ensure this, control variables were implemented to minimise the effects from foreign factors.

Vu Nguyen Le

Control Variable	Why Controlled?	How It Will Be Controlled?
Dataset	To ensure that any differences in model performance are due to the models themselves and not variations in the data.	The same dataset, sentimentDatabase.csv, will be used for training and testing.
Preprocessing Steps	Inconsistent preprocessing can lead to different feature representations, creating different input prompts, which can alter model performance.	The same preprocessing steps (punctuation removal, stopword removal, stemming, lemmatization) will be applied before feeding the dataset to both models.
Training-Testing Split	To prevent different data distributions during training and testing stages and ensure both models are trained and tested on the same volume of data.	An 70-30 split of the dataset will be applied consistently, with the same random seed used for splitting to ensure identical training and testing sets for both models.
Programming Environment (IDE)	Changes in the programming environment, such as settings or restrictions, can introduce variability in execution.	Both models will be programmed, trained, and evaluated on Jupyter Notebook to maintain consistent execution conditions and avoid hardware-specific biases.

Vu Nguyen Le

Hyperparameters (excluding epochs)	Since both models are not of the same ML class, certain hyperparameters can be used to fine-tune the models and improves the performance.	All the hyperparameters specific for both models will be kept at either constant, null, or default based on the provided documentations.

Figure 8. The table of control variables for the experiment.

Vu Nguyen Le

4 Experimental Data

4.1 Raw Data

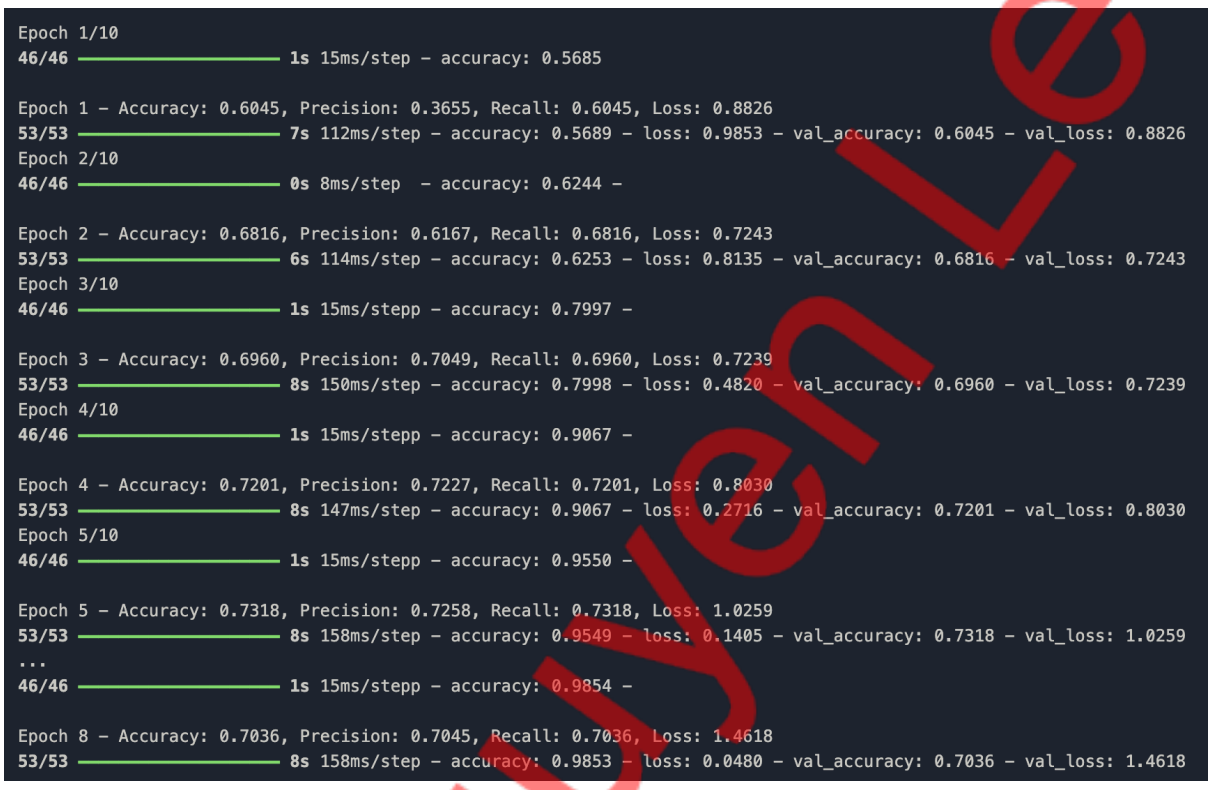


Figure 9. The truncated execution of Bidirectional LSTM model using Keras Layers.

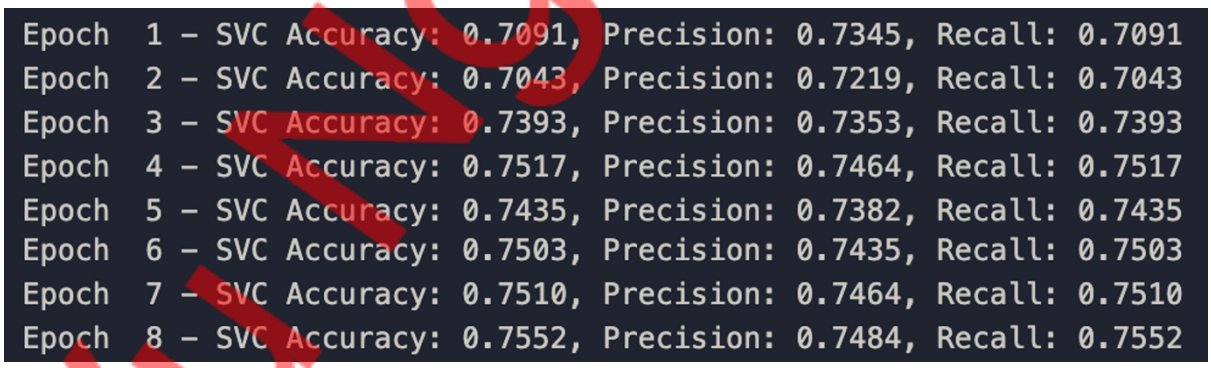


Figure 10. The full execution of the LinearSVC model using Scikit-Learn.

Figures 9 and 10 displays the performance for both models directly from the output terminal. In Figure 9, for each epoch, there is both the training accuracy (calculated based on the training set) and the validation accuracy (calculated based

Vu Nguyen Le

on testing set). For example, for epoch 3, the training accuracy is 0.7997, whilst the validation accuracy is 0.6960. Since the research aims to test the models on unseen data, only the validation accuracy, precision, and recall will be considered for processing the data.

4.2 Processed Data

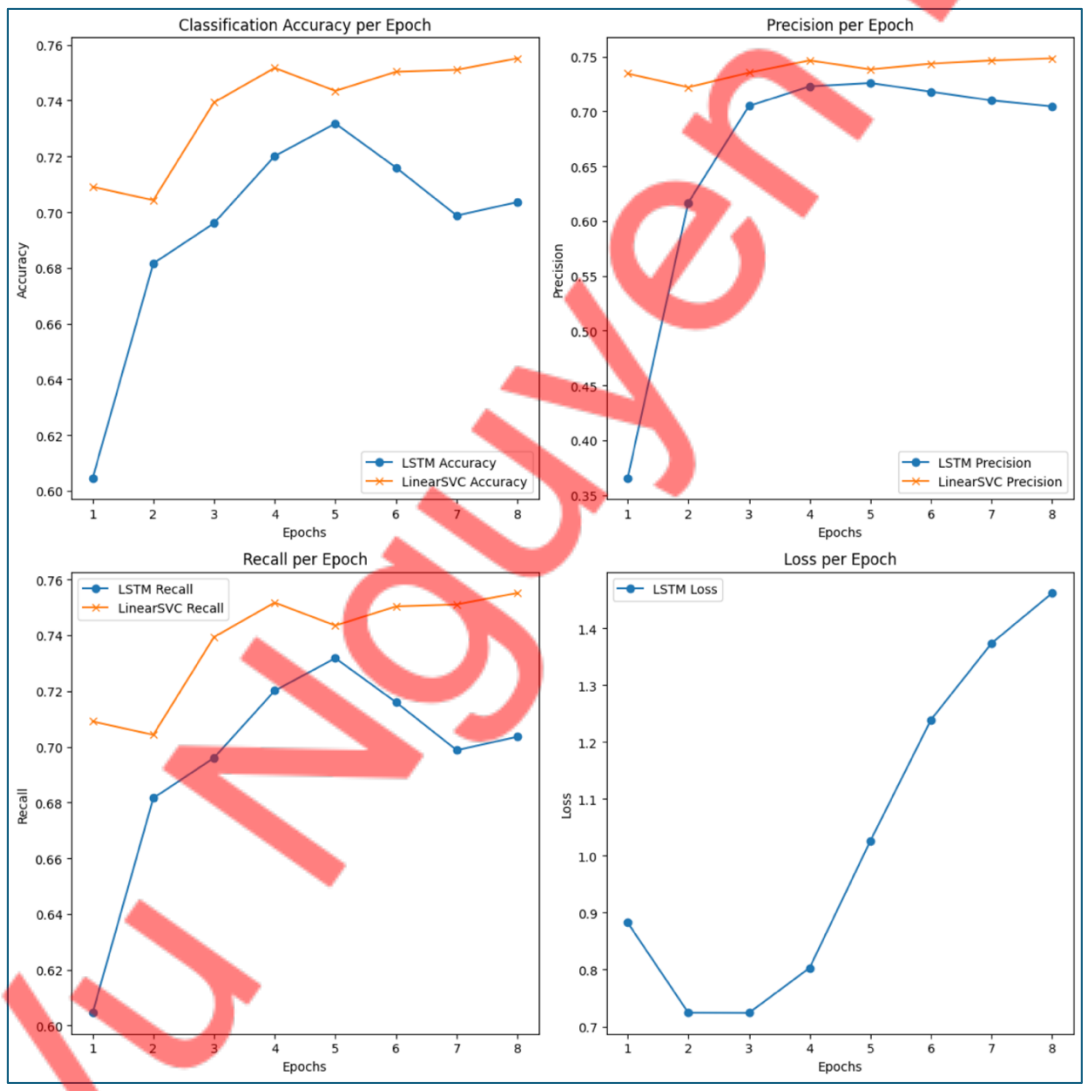


Figure 11. The four plotted graphs of accuracy, precision, recall, and loss (only for LSTM) per epoch during validation (testing on unseen data) of the model.

Vu Nguyen Le

In terms of accuracy, the Bidirectional LSTM initially improves, reaching its peak at epoch 5 with an accuracy of 0.7318, but it declines thereafter, finishing at 0.7036 by epoch 8. In contrast, the LinearSVC starts higher at 0.7091 and continues to outperform the LSTM consistently, stabilising at 0.7552 by epoch 8. For precision, the LSTM increases sharply in the first few epochs, reaching 0.7258 by epoch 5 and then plateauing. LinearSVC remains consistently higher, around 0.73-0.75 across all epochs. Regarding recall, the LSTM also increases sharply and the peak at 0.7318 during epoch 5 before declining to 0.7036 at epoch 8, indicating reduced ability to correctly classify positive cases over time. LinearSVC demonstrates a higher rating throughout than the LSTM. Finally, the loss for LSTM decreases until epoch 3, reaching 0.7239, before increasing sharply to 1.4618 by epoch 8, indicating overfitting after epoch 3.

The final version of the model after the training of 8 epochs were tested and results were visualised using a Confusion Matrix Heatmap and calculated using F1 Score.

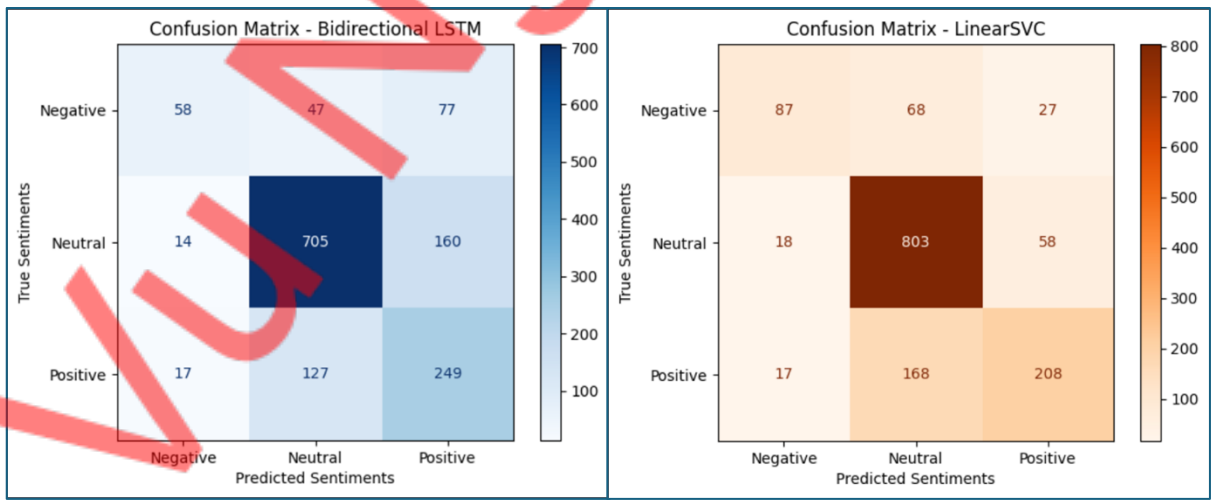


Figure 12. The Confusion Matrix heatmap for both the Bidirectional LSTM and LinearSVC.

Vu Nguyen Le

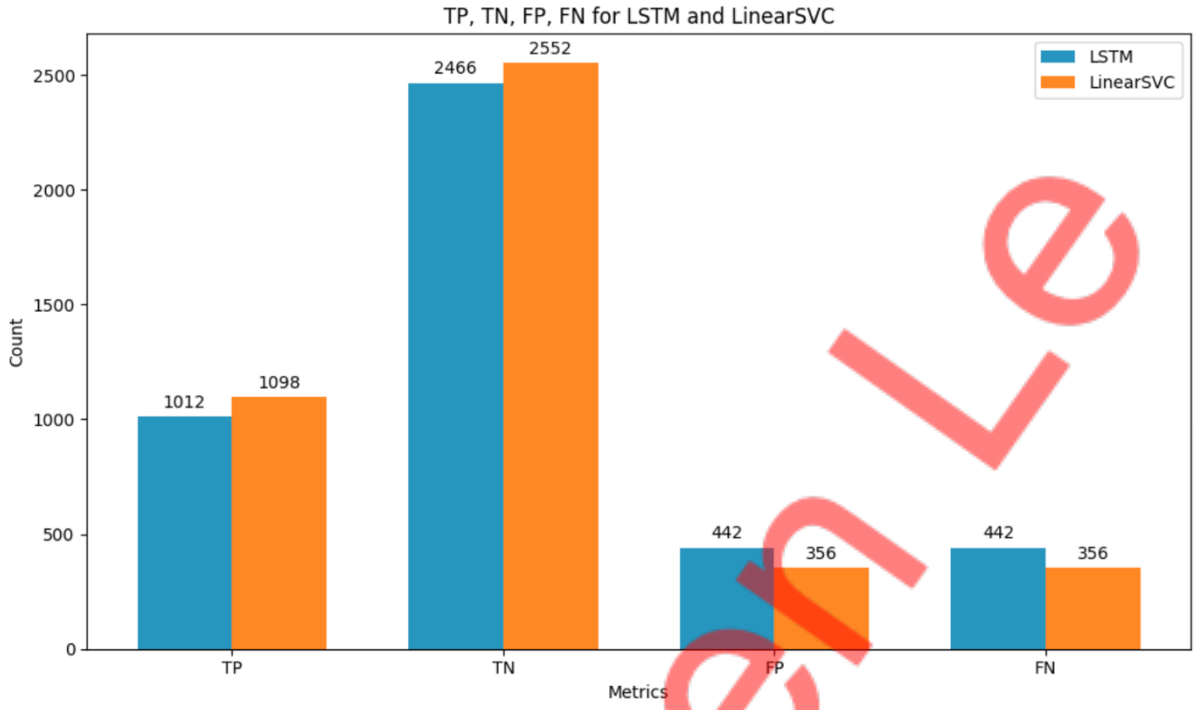


Figure 13. The tally of True Positive, True Negative, False Positive, False Negative for both the LSTM and LinearSVC models.

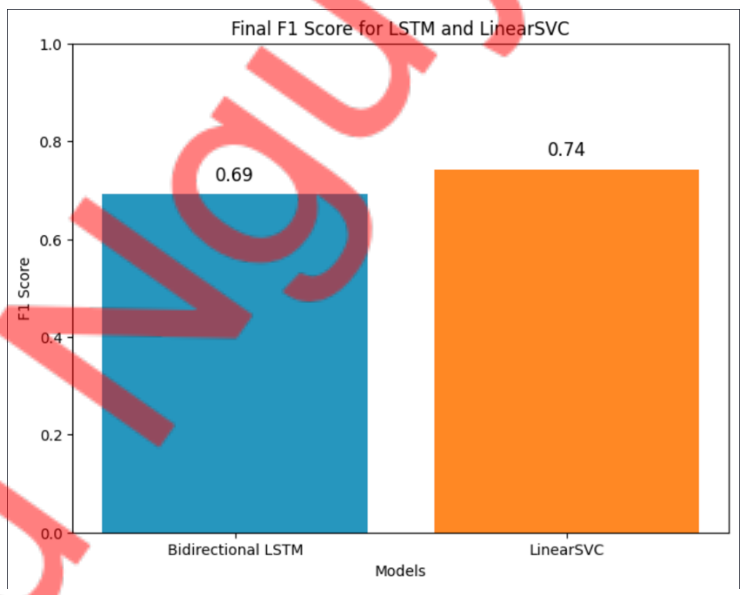


Figure 14. The calculated F1 Score for the final trained version of both models.

Based on the figures, it can be seen that the LinearSVC model demonstrates stronger performance in predicting neutral sentiments, with 803 correct classifications, compared to the LSTM's 705. LinearSVC shows a higher count of true positives (1,098)

Vu Nguyen Le

compared to LSTM's 1,012, while also achieving more true negatives (2,552 vs. 2,466). Both models show an equal number of false positives, but LinearSVC exhibits a lower false negative count (356 vs. 442). This aligns with the overall trend seen across the confusion matrix and classification metrics, where LinearSVC consistently outperforms the LSTM.

5 Conclusion

The result from the experiment disproves the hypothesis made earlier because it was shown that the LinearSVC model outperforms the Bidirectional LSTM in sentiment analysis on financial news headlines. Despite the Bidirectional LSTM's architecture being designed to handle long-term dependencies and complex sequential data, its performance was hindered by overfitting, particularly after epoch 3 (where the loss value suddenly experiences an increasing trend), leading to declining accuracy and recall. In contrast, LinearSVC demonstrated consistent and stable performance across all metrics, with higher classification accuracy, precision, recall, and a superior F1 score. The LSTM, while potentially more powerful when properly tuned, failed to generalise effectively in this experiment due to inadequate regularisation.

6 Evaluation

While the experiment successfully compared the performance for both models, several factors and limitations should be considered when interpreting the results.

Firstly, the Bidirectional LSTM model displayed significant overfitting. While it initially showed improvements in classification accuracy, precision, and recall, performance began to degrade after epoch 3. This suggests that the model started to

Vu Nguyen Le

memorise the training data instead of learning to generalise to unseen data. The application of early stopping helped to some extent, but further regularisation and tuning of hyperparameters, such as learning rate, dropout rates, and batch size, would likely be necessary to prevent this overfitting in future experiments.

In contrast, the **LinearSVC model** performed more consistently across all epochs and metrics. It maintained steady classification accuracy, precision, and recall without any signs of overfitting. This model's performance stability was likely due to its simpler architecture and its ability to generalise effectively despite the absence of sequential data processing capabilities. While the LinearSVC did not fully converge during the training phase, testing with higher iterations (up to 1,000 epochs) showed that this had minimal impact on the overall performance.

Secondly, preprocessing was another critical factor that influenced the models' outcomes. The dataset was subjected to several preprocessing steps, including punctuation removal, stopword elimination, lemmatisation, and stemming. These steps helped reduce noise and standardise input, allowing the models to focus on meaningful patterns within the data. However, the precise impact of these preprocessing steps on each model's performance was not fully explored. Future studies could investigate how varying the preprocessing techniques might influence model outcomes, especially given the LSTM's sensitivity to sequential input. Additionally, the different methods of tokenization (TF-IDF for LinearSVC and Text Vectorization for LSTM) might have contributed to the performance discrepancy and warrant further investigation.

Thirdly, in terms of **control variables**, consistency was maintained throughout the experiment. Both models were trained on the same dataset with identical train-test

Vu Nguyen Le

splits. A 70-30 split was used, and a fixed random seed ensured that both models trained and tested on the same data subsets. Preprocessing methods were also uniformly applied, guaranteeing that any performance differences were due to the models themselves rather than the input data or experimental conditions.

Finally, a key limitation of the experiment was the focus on only two models, the Bidirectional LSTM and LinearSVC. Expanding the study to include other models, such as Transformer-based architectures (e.g., BERT) or ensemble models like Random Forest, would provide a more comprehensive evaluation of the various approaches to sentiment analysis.

Vu Nguyen Le

7 References

Alfarizi, M. I., Syafaah, L., & Lestandy, M. (2022). Emotional text classification using TF-IDF and LSTM. JUITA, 10(2), 225. https://doi.org/10.30595/juita.v10i2.13262

Daniel. (2023, June 2). Epoch: An essential notion in real-time programming. DataScientest. https://datascientest.com/en/epoch-an-essential-notion

Graves, A., & Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional LSTM networks. IEEE Xplore. https://doi.org/10.1109/IJCNN.2005.1556215

IBM. (2023, December 27). What is support vector machine? IBM. https://www.ibm.com/topics/support-vector-machine

Jakubiak, N. (2022). Analysis of explainable artificial intelligence on time series data. Upcommons.upc.edu. https://upcommons.upc.edu/handle/2117/376475

Jozefowicz, R., Zaremba, W., & Sutskever, I. (2015, June 1). An empirical exploration of recurrent network architectures. Proceedings.mlr.press; PMLR. https://proceedings.mlr.press/v37/jozefowicz15.html

Koushiki. (2024, June 13). Sentiment Analysis with LSTM. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2022/01/sentiment-analysis-with-lstm/

Kundu, R. (2022, December 16). F1 score in machine learning: Intro & calculation. V7 Labs. https://www.v7labs.com/blog/f1-score-guide

Lynch, C., O'Leary, C., Smith, G., Bain, R., Kehoe, J., Vakaloudis, A., & Linger, R. (2020). A review of open-source machine learning algorithms for Twitter text sentiment analysis and image classification. IEEE IJCNN.

Vu Nguyen Le

Nielsen, M. (n.d.). *Neural Networks and Deep Learning*. https://static.latexstudio.net/article/2018/0912/neuralnetworksanddeeplearning.pdf

Saeedvand, S. (2023, September 6). CH7 - Machine Learning (ML) - Classification support vector machines (SVM). YouTube. https://www.youtube.com/watch?v=xsZIdDtGElg

Saxena, S. (2021, March 16). LSTM | Introduction to LSTM | Long Short Term Memory. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2021/03/introduction-to-long-short-term-memory-lstm/

Team, Expert.ai. (2022). What is machine learning? Expert.ai. https://www.expert.ai/blog/machine-learning-definition

Zhang, A. (2019). Presidential debate sentiment analysis with LSTM, OneVsRest, LinearSVC: NLP step-by-step guide. Medium. https://medium.com/hackernoon/presidential-debate-sentiment-analysis-with-lstm-onevsrest-linearsvc-nlp-step-by-step-guide-b9683e2c8ed9

Vu Nguyen Le

8 Bibliography

Barney, N. (2023, March). *What is sentiment analysis (opinion mining)? Definition from SearchBusinessAnalytics*. SearchBusinessAnalytics. https://www.techtarget.com/searchbusinessanalytics/definition/opinion-mining-sentiment-mining

codebasics. (2021). Simple Explanation of LSTM | Deep Learning Tutorial 36 (Tensorflow, Keras & Python). In *YouTube*. https://www.youtube.com/watch?v=LfnrRPFhkuY

corinnaabigail. (2020). Sentiment analysis with SVM in Python. GitHub. https://github.com/corinnaabigail/sentiment-analysis-python-with-support-vector-machine

Elite Data Science. (2017, September 7). *Overfitting in Machine Learning: What It Is and How to Prevent It*. EliteDataScience. https://elitedatascience.com/overfitting-in-machine-learning

Nielsen, M. (2017, June 29). Neural networks: representation. Jeremy Jordan. https://www.jeremyjordan.me/intro-to-neural-networks/

OpenAI. (2024). ChatGPT. https://chat.openai.com/chat

Schölkopf, B., & Smola, A. J. (2002). Learning with kernels: Support vector machines, regularization, optimization, and beyond. MIT Press.

Yu Nguyen Le

Sentiment analysis with LSTM. (2022, January 17). Analytics Vidhya. https://www.analyticsvidhya.com/blog/2022/01/sentiment-analysis-with-lstm/

shwethaspdevam. (2020, July 13). COMPARISON OF MODELS FOR FINANCIAL NEWS. Kaggle.com; Kaggle. https://www.kaggle.com/code/shwethaspdevam/comparison-of-models-for-financial-news

Vapnik, V. N. (2000). *The Nature of Statistical Learning Theory*. Springer New York. https://doi.org/10.1007/978-1-4757-3264-1

Vu Nguyen Le