[1] Shubham Ojha
[2] Glenn Mendonca
[3] Glen Rodrigues
[4] Kranti Wagle
[5] Ashwini Pansare
[6] Kalpana Deorukhkar

Enhancing Stock Market Predictions with a Stacked CNN-LSTM Ensemble Model

Abstract: - The prediction of stock market movements remains a complex and challenging task due to the dynamic and intricate nature of financial markets. In recent years, deep learning models have emerged as promising approaches for capturing patterns and dependencies in financial time series data. This research paper presents an innovative approach to stock market prediction using a stacked ensemble consisting of CNN-LSTM networks, for multiple tasks.

The proposed Stacked CNN-LSTM Ensemble (SCLE) model leverages the unique strengths of CNNs and LSTMs to effectively learn hierarchical representations from historical stock market data. The CNN component focuses on extracting local features and capturing short-term dependencies by operating on the raw input sequences. Meanwhile, the Bidirectional LSTM component models the long-term dependencies and temporal dynamics inherent in the sequential data, both from front to back, and vice-versa. Such is the ubiquity of this model architecture, that it shows state- of-the-art performance for both time-series prediction as well as sentiment analysis.

Training data was collected by methods of scraping news articles for sentiment analysis and using Python libraries to get the latest closing value of a particular selected stock.

To evaluate the performance of the proposed SCLE model, extensive experiments were conducted, to see its performance in capturing the historic trends. Multiple companies from the Fortune 500 list were chosen for this task. The results demonstrate that the proposed ensemble outperforms individual CNN and Bidirectional LSTM models, as well as other commonly used baseline methods for stock market prediction. The SCLE model achieves higher accuracy, robustness, and generalization capabilities, resulting in improved forecasting of stock market movements, using both historical data as well as current market sentiment analysis.

Overall, this research paper introduces a novel ensemble model, the SCLE model, which combines CNNs and Bidirectional LSTMs for stock market prediction. The proposed Ensemble model is an attempt at generalizing the predicted value, to incorporate not only temporal stock trends but also quantize current market sentiment, into a meta-model that has the potential to outperform other state-of-the-art models.

Keywords: Stock market prediction, Stacked Ensemble model, CNN-LSTM, Sentiment Analysis, Regression models

I. INTRODUCTION

The stock market forms the backbone on which world economies thrive and function. Regardless of its complex nature, unpredictability, and high non-linearity, it has managed to garner interest around itself from investors and corporations alike, because of its high-return characteristics. This gave impetus to growing methods of research, aimed toward accu- rate and reliable methods for predicting stock prices. Another reason for the same would be that potential investors would prefer to make more informed decisions, and thereby reduce any potential risk associated with investing in the stock market. Recent advances in computational processing and innovations in the field of Deep Learning have allowed researchers to model stock market trends and analyze them in greater detail. Neural Networks have been widely considered to fit well with non-linear data. Recurrent Neural Networks, in particular, have been extensively used to model, analyze and predict sequential data, and time-series analysis, much like the data in which stock values are available. In this paper, we propose a Stacked Ensemble Attention-Based Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) & Bidirectional LSTM model for stock price prediction.

[1] Dept of Computer Engineering, Fr. C. Rodrigues College of Engineering, Mumbai, India. Email: crce.9276.cs@gmail.com

[2] Dept of Computer Engineering, Fr. C. Rodrigues College of Engineering, Mumbai, India. Email: crce.9272.cs@gmail.com

[3] Dept of Computer Engineering, Fr. C. Rodrigues College of Engineering, Mumbai, India. Email: crce.9287.cs@gmail.com

[4] Dept of Computer Engineering, Fr. C. Rodrigues College of Engineering, Mumbai, India. Email: kranti@fragnel.edu.in

[5] Dept of Computer Engineering, Fr. C. Rodrigues College of Engineering, Mumbai, India. Email: ashwinip@fragnel.edu.in

[6] Dept of Computer Engineering, Fr. C. Rodrigues College of Engineering, Mumbai, India. Email: kalpanas@fragnel.edu.in

The proposed model combines the outputs of two individual models, into a meta-model. The first model is an Attention- Based CNN-LSTM Model, for time-series prediction of a stock using historical stock prices and over 87 technical analy- sis values. The second model is a Bidirectional LSTM, which does sentiment analysis on news articles scraped from the internet, company-wise, and classifies them into (-1) Negative Sentiment, (0) Neutral Sentiment, and (1) Positive Sentiment. The output from both these models is their predictions of the stock value, 5 days in the future. These predictions are then sent to the final model that gives as output, more informed predictions, relevant to the current market sentiment. The final model will be trained on the outputs of the two models as independent variables, and the actual stock value (on that day) as the dependent variable. Thus, on receiving new predictions from the models, the final Regressor Model will output the final predicted 5-day values.

The efficacy of the proposed model was evaluated on the stock market dataset taken from Yahoo's open source Library, yfinance, and performance was compared with that of other state-of-the-art stock market prediction models. The results demonstrate that the proposed hybrid model outperforms other methods in prediction accuracy, highlighting the potential of combining other multiple sources of information for predicting the stock market. This study provides valuable insights into the use of hybrid attention mechanisms in stock market prediction and opens up avenues for future research in this area. We believe that the proposed method will greatly interest researchers, practitioners, and investors alike. In the rest of the paper, we provide a detailed description of the proposed model, including the sentiment analysis component and the attention mechanism. We also describe the evaluation method and results of the study, as well as the limitations and future work.

Overall, this paper provides a comprehensive and in-depth analysis of the use of a Stacked CNN-LSTM Ensemble for Stock Market Prediction.

II. RELATED WORK

Numerous studies have been done on Stock Market fore- casting, each introducing a unique and different methodology. This section summarizes some of the research and literary work performed in this area.

[1] paper discusses the difficulty of predicting stock market prices and the correlation between news articles and stock price movements. To improve accuracy, the authors gather a large dataset of daily stock prices for S&P 500 companies for five years, along with over 265,000 related news articles, and analyzed it using deep learning models. Cloud computing is used for training and real-time inference. Index terms include stock market prediction, cloud computing, big data, machine learning, and regression.

[2] goes into detail on how predicting stock prices accurately is difficult due to various factors, making data analysis a potential game changer. Historical stock prices carry the impact of all events, so machine learning techniques are employed to infer future trends. A framework using the LSTM model and net growth calculation algorithm is proposed for the analysis and prediction of a company's future growth.

Hamoudi et al., [3] from Stanford University, aim to develop 1-Dimensional CNN and LSTM prediction models for high- frequency algorithmic trading using a data set of 130 intra-day market features and trade returns. Binary classification is used, with a logical matrix augmenting missing data. Three LSTM and two CNN candidate models are compared, with out-of- sample test results showing high average return per trade and low overall risk.

Nabipour et al., [4] from Cornell University, focuses on predicting future values of four stock market groups from the Tehran stock exchange using machine learning algorithms.

The algorithms include decision tree, bagging, random forest, Adaboost, gradient boosting, XGBoost, ANN, RNN, and LSTM. LSTM showed the most accurate results, and for tree- based models, Adaboost, Gradient Boosting, and XGBoost were competitive. Technical indicators were used as inputs, and results were presented based on four metrics.

[3] is another study from Stanford, that presents a long-short- term memory (LSTM) model for predicting market behavior and is used to develop a StockBot for buying/selling stocks to maximize profits. The model leverages the sequential structure of time-series data and outperforms the market with gains 15 times higher than the most aggressive ETFs.

[4] examines whether sentiment in news articles can predict daily stock market movements. A dataset from 2013 to early 2020 was created, using news articles and S&P 500 price movements. Different feature sets and machine/deep learning models were compared, finding that stock market movements can be predicted using news sentiment polarity and technical analysis features. The proposed support vector machine model resulted in promising outcomes, allowing investors to make better decisions about buying or selling stocks.

[5] proposes a deep learning method based on a Convolutional Neural Network (CNN) to predict the stock price movement of the Chinese stock market. The input is derived from the internet and includes the opening price, high price, low price, closing price, and volume of the stock. Results indicate the method is somewhat reliable for predicting stock price movement in China.

The project in [8] explores the relationship between financial news articles and stock trends by using news sentiment classification. Three different classification models were created and tested, with RF and SVM performing well. The prediction model showed an accuracy of more than 80% and increased the accuracy by 30% compared to news random labeling.

[9] aims to predict stock market movements by combining historical prices with sentiment data. Two models were used: LSTM with historical prices and Random Forest with sentiment analysis and macro parameters. The study predicted the prices of four stocks and evaluated the results using the RMSE metric.

The work in [10] presents an attention-based CNN-LSTM and XGBoost hybrid model for stock price prediction, which integrates traditional time series models with deep learning architectures. The model uses convolution and LSTM to extract deep features and long-term time series features, followed by XGBoost for fine-tuning. Results show that the hybrid model is effective and can help investors make decisions. The source code is available on GitHub. [16]

[11] utilizes Artificial Neural Network and Random Forest to predict the next-day closing price of five companies using financial data. New variables are created using Open, High, Low and Close prices as inputs. Evaluation of models using RMSE and MAPE shows that they are efficient in predicting stock closing prices.

[12] explores the use of four types of deep learning architectures, including Multilayer Perceptron (MLP), Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTMs), and Convolutional Neural Networks (CNNs), to predict the stock price of a company based on historical prices. The models were trained with the stock price of a single company from NSE and predicted for five different companies from both NSE and NYSE. The results show that CNN outperforms the other models and ARIMA, indicating the effectiveness of neural networks in stock price prediction.

[13] describes how the Indian stock market has been significantly impacted by internet-based technologies over the last 20 years, allowing investors to buy and sell shares from anywhere at any time. With the rise of digital technologies, sentiment analysis or opinion mining has become an important tool for predicting stock market indicators such as Sensex and Nifty. This research work focuses on the importance of sentiment analysis for stock market prediction and provides suggestions for future work.

[14] examines the accuracy of selected machine learning algorithms for sentiment analysis and prediction of the Indian stock market during the Covid-19 pandemic. Six algorithms are tested, and the study identifies the most accurate ones, which can be used as inputs for building robust prediction models.

[15] study proposes the use of machine learning algorithms for sentiment analysis of social media data to predict stock market prices. The study utilized Support Vector Machine for sentiment analysis with 83% accuracy and Artificial Neural Network for stock price prediction with a mean squared error of 5.6. This research provides evidence that sentiment analysis can be incorporated into stock price prediction.

III. BACKGROUND

A. Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) are a subclass of deep neural networks that are frequently employed for clas- sifications based on many input features, typically numerical data. CNNs use a combination of convolutional layers, pooling layers, and fully connected layers to learn and extract features from input datasets automatically.

The convolutional layer, which conducts the convolution math- ematical operation, is a crucial part of a CNN. A tiny filter or kernel is slid across the input data during convolution, and a dot product is performed between the filter and each feature at each place to create an output feature map. The network can now automatically find regional patterns or characteristics in the input data thanks to this technique. After convolutional layers, pooling layers are frequently applied to lower the spatial dimensions of the feature maps while maintaining crucial information. By averaging or employing maximum pooling, pooling involves lowering the resolution of the feature maps by summing each local neighborhood of data.

The actual classification or regression work is then carried out by mapping the retrieved features to the output labels using fully connected layers.

B. Long-Short Term Memory Networks (LSTM Networks)

The Long Short-Term Memory (LSTM) architecture for recurrent neural networks (RNNs) was created to address the drawbacks of conventional RNNs, which frequently experi- ence the vanishing gradient problem. LSTM networks are frequently used to handle sequential data, including time series data, audio, and text written in natural language.

The input gate, forget gate, and output gate are the three main gates found in an LSTM cell, which is the main part of an LSTM. The LSTM can selectively recall or forget information over time thanks to these gates, which regulate the information flow across the cell.

The LSTM cell gets a series of input values one at a time during training and generates a series of output values in response. Each time a time step is reached, the input value and the previous hidden state are combined to create a new candidate value. This new candidate value is then filtered and altered by the gates to create the final output and the new hidden state.

The forget gate controls how much of the prior cell state should be kept, while the input gate defines how much of the new input value should be added to the cell state. After that, the output gate decides how much of the updated cell state should be used to generate the output value. The capacity of LSTM networks to selectively recall or forget information over extended periods of time is one of its main advantages. This qualifies them for sentiment analysis of news articles pertaining to a stock. This improves stock price forecasting even further.

C. CNN-LSTMs

CNN-LSTM models can be used for time series prediction and sentiment analysis by combining the strengths of both Convolutional Neural Networks (CNNs) and Long Short- Term Memory (LSTM) networks. CNNs are useful for feature extraction, while LSTMs are effective for handling sequences and capturing long-term dependencies in the data.

In time series prediction, CNN-LSTM models can help un- cover hidden structures in the input signals, as mentioned by a user in machinelearningmastery.com. The CNN layer can be used to extract features from the input, and the LSTM layer can model the temporal dependencies of the extracted features. However, it is essential to note that LSTMs may not always outperform classical time series methods or other machine learning models, and it's recommended to try different approaches.

For sentiment analysis, CNN-LSTM models can be applied to text data, where the CNN layer can capture local features like n-grams, and the LSTM layer can model the sequential information in the sentences. This combination allows the model to learn both local and global features in the text, leading to improved sentiment classification performance.

In general, CNN-LSTM models work by first passing the input data through a series of convolutional layers to extract meaningful features. This process involves convolving the input with a set of filters or kernels, followed by activation

functions and pooling operations. The extracted features are then fed into the LSTM layer, which captures the temporal or sequential dependencies within the data. Finally, the output of the LSTM layer is passed through a dense layer for prediction or classification

IV. PROPOSED METHODOLOGY

This section provides an in-depth description of our method- ology. It includes an end-to-end design of a stock-market prediction system, from data aggregation and collection to the final output.

A. Data Collection

The proposed methodology requires data from two sources, historical stock values and average sentiment about the cor- responding company. The former is achieved with the help of Yahoo's finance [17] library, while the latter involves a novel system designed to scrape news articles, and perform sentiment analysis on it. The following are the methods described in detail.

1) Historical Stock Values: Yahoo's yfinance package is a popular Python library designed for retrieving financial data from Yahoo Finance. It allows users to access historical stock prices, market data, and financial reports simply and efficiently. The package is widely used by finance professionals, data analysts, and developers for various purposes, such as analyzing stock trends, building financial models, or creating trading algorithms. You simply need to specify the stock's name and the period for which you need the stock values. It returns a pandas

DataFrame of 6 values: Open, Close, High, Low, and Volume Traded. After repeated iterations, we found that using these values alone did not give enough insights into the model and that the predictions were way off. Thus, incorporating more financial knowledge into the model was the next step. Technical analysis values are quantitative measures that are used in the field of technical analysis to evaluate and analyze financial markets, securities, and assets. These values are calculated using historical price and volume data to identify trends, patterns, and signals that can be used to predict future price movements. Python provides another package for this purpose, called ta [18]. It returns a total of 86 technical analysis values, on giving the details about a stock. The final pandas dataframe sent to the time series model, thus has a total of 87 values.

2) Sentiment Values: The second part of the methodology requires sentiment analysis values for each company. The entire system is managed using Python's Django framework [22]. This is achieved by making use of the Scrapy Python package to perform parallel site scraping utilizing multiple system threads. The stocks to be analyzed are fed through Django based on user requirements, and URLs corresponding to the specified stocks are generated from seed URLs of the Economic Times, Money control and Live Mint base address. The latest articles related to the stock are fetched and necessary features such as title, description, article, publish date, etc. are scrapped using libraries like Beautiful Soup, LXML, etc. Every article has a determined HTML DOM structure which can be traversed using XPaths. The scrapped articles are prepressed to remove redundant symbols and ASCII characters and the generated data is stored in the SQL database for further processing of market sentiments. The news articles are scrapped periodically in 6 hours to get the latest stock views to reduce system load and minimize the scrapping of the same articles. The Scrapy engine is scheduled using the django_cron library and scrapes on 32 concurrent URLs, however, it highly depends on the system resources.

After the collection of news articles, they were labeled using a pre-trained transformer, FinBERT [19], which is trained to tackle NLP tasks in the financial domain.

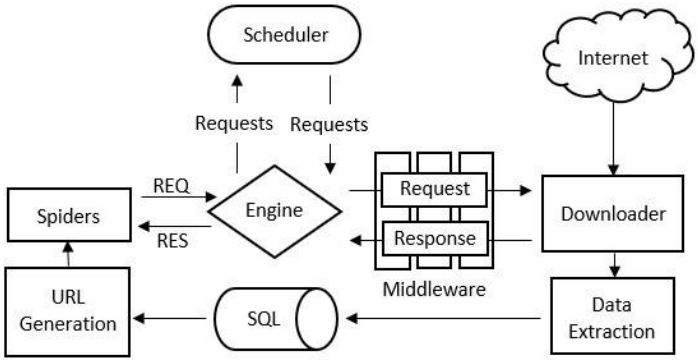

Fig. 1. Scraping Methodology.

B. Database Schema

The database has 6 tables that work together to store data in 3rd normal form. The cronjob collects data from various sources before the preprocessing begins. The collected data is validated and stored in the database. The sentiment analysis models fetch news articles one by one from the database and output the news sentiment back into the database. This process is slow and time-consuming, however, the database commits can be improved to speed up this process. The sentiment data stored uses various transformations to output the required data for the next 2 models. The historical model uses the past stock values stored in our database and the sentiment model uses the sentiments stored in the database. All preprocessing models are database intensive, however, saving the data to the database allows us to run the cron algorithm once every week and render the website faster.

C. Data Preprocessing

Any Machine Learning-based system runs on data, one that it has been trained on, and one that it receives when deployed. But data in its raw form is unclean, non-uniform, and inconsistent. This makes Data Preprocessing the most important step of Machine Learning. Unless the data is clean, uniform, and consistent, no model would be able to produce comprehensible results. Our methodology, having two models, requires two separate methods of data preprocessing. It must be noted that all these steps are implemented using sklearn's data pipeline.

1) Historical Data: As mentioned earlier, Yahoo's yfinance provides stock value data for any given stock's name. Since these values are stock data, also referred to as time series data, they are bound to be clean, uniform, and consistent. Thus, there is little scope for data cleaning. However, such data still needs to be transformed.

First, we add technical values to the already present stock data and drop the Open, High, Low, and Volume columns. This makes the final input DataFrame have 87 values in total. After this, the values are Scaled using RobustScaler. The formula for the same is:

This is

$$x_{scaled} = \frac{x - \text{median}(x)}{\text{IQR}(x)}$$

where, IQR = Inter Quartile Range.

This scaling is applied, so that the data is uniformly distributed, and more suited to be given as input to Neural Networks, as opposed to values spanning across a large scale. Since our methodology involves the prediction of stock values of 5 days in the future, we need to prepare the input data as such. This is done by taking the stock values as input into a function with parameters (input_sequence, past_len, future_len), where past_len is the number of days in the past that the model will look to learn all local and global patterns in the stock data. Similarly, future _len takes the constant value of 5, which is the number of days of prediction. Thus the data is in the form, of 1000 days' values as the independent variable and the future 5 days' values as the dependent variable.

2) Sentiment Data: As shown in Figure 1, News articles are scraped by generating URLs for the required stocks and stored in a SQL Database, PostgreSQL. As they are being stored, a Sentiment Analysis Model described in Section E runs on it, and stores the sentiment value for the article, in the corresponding column, titled sentiment _val in the database. Preprocessing data for tasks of NLP is crucial since textual data cannot be directly fed to a model. Thus the general steps include tokenization, lemmatization or stemming, creating embeddings, padding, and finally splitting into train and test datasets. A similar process is followed in our methodology. Each article is first passed to a function to clean the text. All characters are lowered, all non-alphanumeric characters are removed, and all words are split based on whitespaces. After that, we make use of nltk's [21] stopwords list, which contains a set of most commonly occurring words in the English language, that do not add much meaning to the sentence, and the semantic meaning of the sentence remains unchanged even if they are absent. Finally, to create embeddings, we use sklearn's Countvectorizer, which converts a collection of text documents into a sparse matrix of token counts. The formula for the same is:

$$\text{Count}(d, t) = \sum_{i=1}^{|d|} \delta(d_i, t)$$

After this, we simply do a padding on all the vectors created, so that they are all of a fixed length of 5000 elements. Padding is done by appending the required number of 0's to the vector of embeddings for the respective article.

To make the data ready for the prediction of stock values for the next 5 days, the sentiment values for each company, are stored in a list of lists, each having 5 values. Corresponding to the average sentiment of a stock for a particular day, the stock value for the particular day is stored to create a pandas time- series DataFrame of two columns, daily average sentiment value, and stock value for that day. In case there are no news articles for a particular stock for more than 14 days, the average sentiment is adjusted to be neutral i.e. 0.

As part of the final step, the model just needs the most recent sentiment value, a value from -1, 0, 1, which is used to calculate the average sentiment value, and as a result, the model outputs the prediction for the future 5 days.

D. System Design & Workflow

Our novel system architecture is presented in Figure 2. It combines two separate models and takes their predictions as inputs to a third model, for the final prediction. The first component is a time series prediction model, which takes as input historical stock values and their corresponding technical analysis values, and gives as output its 5-day prediction.

The second component is slightly more complicated. It involves scraping finance news articles for a company and doing sentiment analysis on them, in order to quantify the company's current market sentiment. We pair these daily average sentiment values, with the corresponding stock value for that day. The detailed data preprocessing steps are mentioned in the previous section. The final output of this component, using a Regressor model (such as Random Forest Regressor) is a 5-day prediction as well.

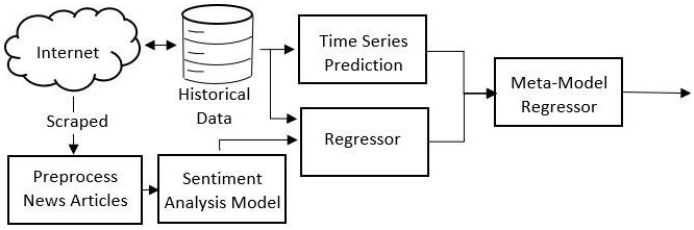

Fig. 2. System Workflow

The outputs of each of the two models are then sent to the final component, which is another Regressor model (such as Support Vector Machine or XGBoost). The independent vari- ables are the previous predictions of the first two components, and the dependent variable is the corresponding actual stock value. This final component can also be called a Meta-Model.

Thus, the Meta-Model gives a more informed and precise 5- day prediction.

E. Visualization

The dataset used for training the sentiment analysis model had some interesting insights that were found on performing Exploratory Data Analysis (EDA). Figure 3, below shows the distribution of the training data, which was found as a list of curated Indian articles, and labeled using FinBERT. The figure shows a slightly larger distribution of negative and neutral articles, than positive articles.

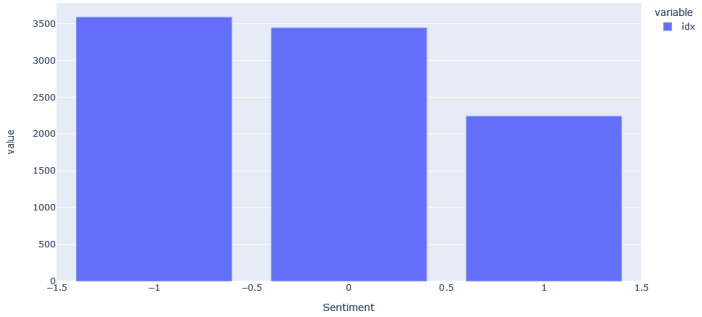

Fig. 3. Distribution of Data

Figure 4 shows the Box Plot and 5-number summary for the article lengths from the training data. As evident, the median length of the articles is about 350 characters, with a few outliers being as long as 7000 characters.

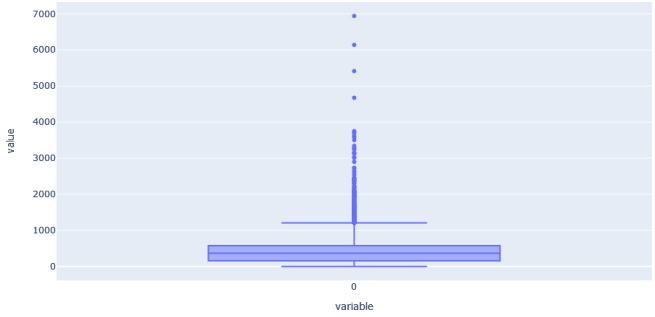

Fig. 4. Box Plot of Article Length

Figure 5 shows the distribution of text length for all positively labeled articles, along with the corresponding 5-number sum- mary, similarly, Figures 6 and 7 do the same for neutral and negative classes, respectively.

Figures 8, 9, and 10 present the Word Cloud for each sentiment class for our sentiment analysis problem.

Figure 8, the word cloud for positive articles has the 100 most frequent words in the corpus. As is evident, words that reinforce a sense of affirmation and confidence result in the article being classified as positive. Some of these words are

	Length
count	4390.0
mean	431.98
std	379.92
min	0.0
25%	158.0
50%	365.0
75%	573.0
max	6140.0

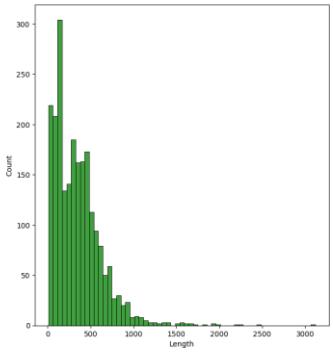

Fig. 5. Text Length Distribution & 5-number summary of Positive articles

	Length
count	3445.0
mean	511.32
std	457.87
min	0.0
25%	184.0
50%	422.0
75%	711.0
max	6140.0

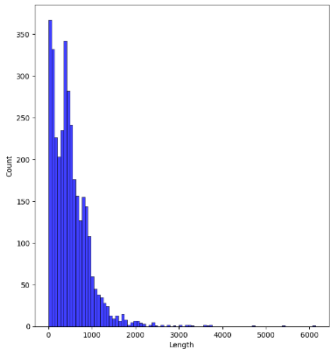

Fig. 6. Text Length Distribution & 5-number summary of Neutral articles

growth, market, increase, crore, global, etc. Figure 9, the word cloud for negative articles has the 100 most frequent words in its respective corpus. These are words that make investors and the general public lose confidence in the market, resulting them in pulling their money off of it, resulting in a down market. Some of these words are government, would, market, people, China, etc. Figure 10, the word cloud for neutral articles has the 100 most frequent words in its respective corpus. These are words that have a more or less similar impact on the

	Length
count	4891.0
mean	431.38
std	375.23
min	1.0
25%	156.0
50%	366.0
75%	580.0
max	6942.0

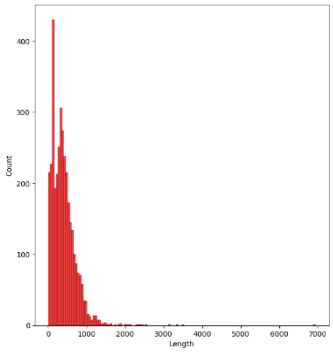

Fig. 7. Text Length Distribution & 5-number summary of Negative articles

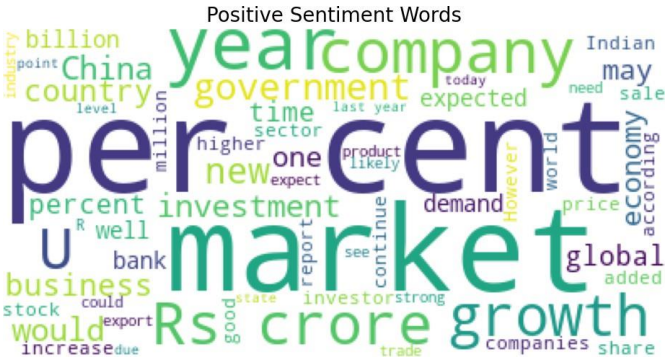

Fig. 8. Positive articles' Word Cloud

Fig. 9. Negative articles' Word Cloud

Sentiment of the article. Interestingly, as seen in Figure 3, neutral and negative sentiment articles are more in number than positive ones. Thus, the most frequent words in this corpus are government, China, year, percent, etc.

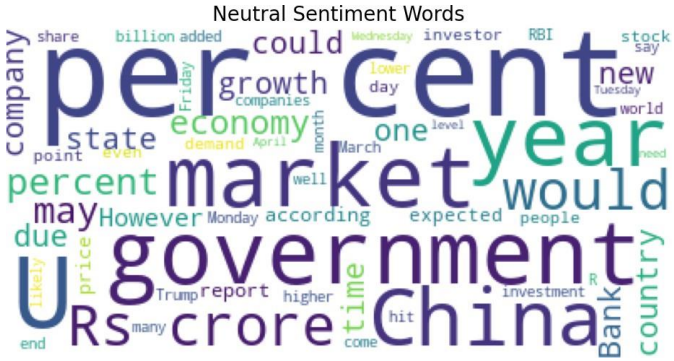

Fig. 10. Neutral articles' Word Cloud

F. Algorithms & Model Analysis

Multiple Deep Learning algorithms for Time-Series Predic- tion were tested, for all models, one using historic stock values, sentiment analysis, and final prediction.

Table I gives a summary of the first model, which does Time Series Prediction using Historic values. All models were

trained for 100 epochs, using the Adam Optimizer [20] with its default learning rate of 0.001, with a batch size of 32, a validation split of 20%, and an early stopping patience of 20 on the validation loss. These parameters were chosen after multiple experiments. The stock data considered was 10 years' stock data of the Indian IT company, Tata Consultancy Services (TCS).

The first model to be chosen were Recurrent Neural Networks (RNNs), as they are the default first choice for tasks involving sequential data. However, the exploding gradient problem becomes apparent quite early, due to the magnitude of our data. The exploding gradient problem in recurrent neural networks (RNNs) occurs when the gradients computed during back- propagation become excessively large. This leads to unstable learning dynamics and difficulty in converging to an optimal solution. The main cause of the exploding gradient problem is the multiplication of gradients as they are backpropagated through multiple layers of the network. When the gradients

are greater than 1, their repeated multiplication can quickly lead to exponentially large values. This issue is particularly prevalent in deep neural networks with many layers, where the effects of the exploding gradients can accumulate. The consequences of the exploding gradient problem include slow convergence or failure to converge altogether, as well as unstable or oscillating behavior during training. The network may struggle to generalize well to new data and exhibit poor performance.

LSTMs incorporate a memory cell, input gate, forget gate and output gate. The gates control the flow of information, allowing the LSTM to selectively retain or discard informa- tion. The memory cell preserves long-term dependencies and avoids interference from exploding gradients. This mechanism enables LSTMs to effectively capture and process sequential data without suffering from unstable learning dynamics.

LSTMs are complemented well when a Convolutional Neural Network (CNN) component is attached on top of them. CNNs excel at extracting spatial features from data, such as images, by leveraging convolutional filters to capture local patterns, using pooling, padding, and strides. On the other hand, LSTMs are designed to capture temporal dependencies in sequential data by modeling long-term dependencies and preserving information over time. By combining both CNNs and LSTMs, the CNN-LSTM architecture can capture both spatial and temporal features, making it suitable for analyzing complex and dynamic data, such as stock market prices. As seen, CNN-LSTMs give the best results.

TABLE I. Time-Series Model Architecture comparison

Model Architecture	**Metrics**			
	Accuracy	Loss	Validation Loss	Recall
RNN	0.2541	0.0254	0.0361	0.3014
LSTM	0.2642	0.0254	0.0339	0.3039
GRUs	0.2316	0.0255	0.0339	0.3033
Auto-Encoder	0.3145	0.0219	0.0226	0.412
CNN-LSTM	0.4325	0.0104	0.0183	0.5021

Table II summarizes the algorithms used for Sentiment Analy- sis of News Articles. A number of Machine Learning & Deep Learning Algorithms were used. Owing to the long length of the articles, CNN-LSTMs were considered here too, and they outperformed other algorithms. A similar argument is valid in this case as well.

Table II. Sentiment-Analysis Algorithms & Model Architecture comparison

Algorithm	**Metrics**				
	Accuracy	Precision	Recall	F1 Score	Loss
Naive Bayes	0.75	0.80	0.70	0.74	-
SVM	0.76	0.78	0.75	0.76	-
Random Forests	0.81	0.84	0.80	0.82	-
XGBoost	0.85	0.86	0.84	0.85	-
LSTM	0.87	0.88	0.87	0.87	0.30
CNN-LSTMs	0.92	0.90	0.89	0.89	0.18

Table III summarizes the Regression algorithms used to measure the accuracy of the final meta-model, the one that gives as output the final value of our system.

Table III. Regression Algorithms for the final prediction, using the meta-model

Algorithm	**Metrics**			
	MSE	MAE	MAPE	R2
Decision Tree	0.00031	0.042	0.0021	0.91
Support Vector Machine	0.0026	0.0038	0.0019	0.92
k-NNs	0.0032	0.043	0.0022	0.90
Random Forest	0.0023	0.036	0.0018	0.93

The final model selected for the meta-model is Random Forest. However, even more powerful and robust algorithms like Gradient Boosting can be considered, once a large amount of data is gathered.

B. Evaluation Metrics

This section gives a summary of all models that were finalized as a result of comparisons of multiple models & algo- rithms. First, we begin with the CNN-LSTM Model used for Time Series Prediction. Figures 8 and 9 below show the plots for Training and Validation Loss & Accuracy respectively. As mentioned previously, training for the models was specified at 100 epochs, with an early stopping parameter of 5 epochs, in order to prevent overfitting.

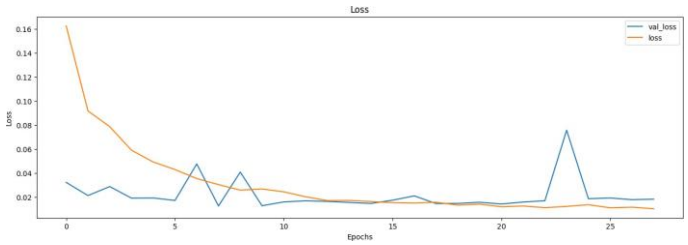

Fig. 11. Training plots for Validation & Training Loss

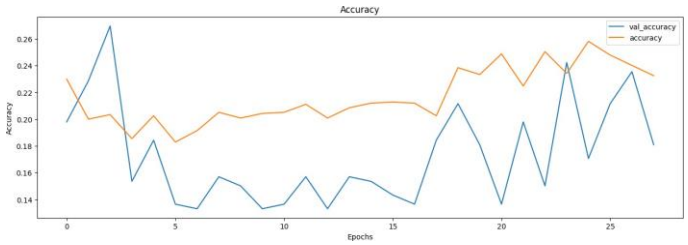

Fig. 12. Training plots for Validation & Training Accuracy

Next, we come to the evaluation plots for the CNN-LSTM Model for Sentiment Analysis. Figure 10 shows the respective plots for Accuracy & Loss plots involving both train and validation datasets. Figure 11 shows the performance of the

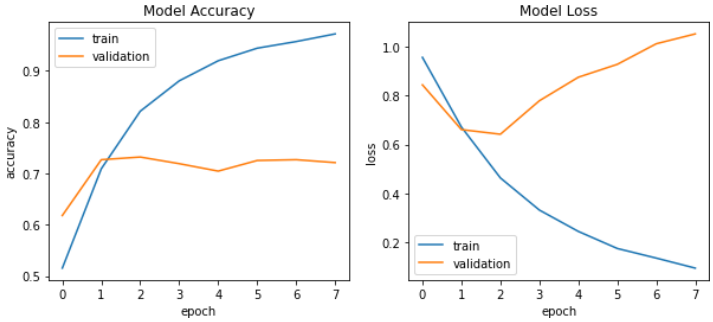

Fig. 13. Training plots for Validation & Training Accuracy

model in predicting the future value of the stock 'TCS.NS'. Note that this is only the output of the Time Series Prediction model, and the final prediction values are different. Figure 12

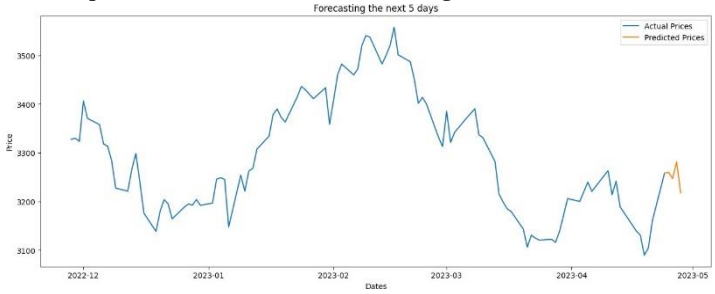

Fig. 14. Prediction of 5-day future stock value

shows the confusion matrix for all three classification classes, Positive, Negative & Neutral Articles. As is evident, most articles are correctly classified. Positive articles being slightly lesser in number in training and testing sets seem to be a little misclassified. This can be corrected by using methods of Validation, such as Stratified K-fold Cross-validation.

V. CONCLUSION

In conclusion, this research paper introduced a novel Stacked CNN-LSTM Ensemble (SCLE) model for stock market prediction. By combining the strengths of Convolutional

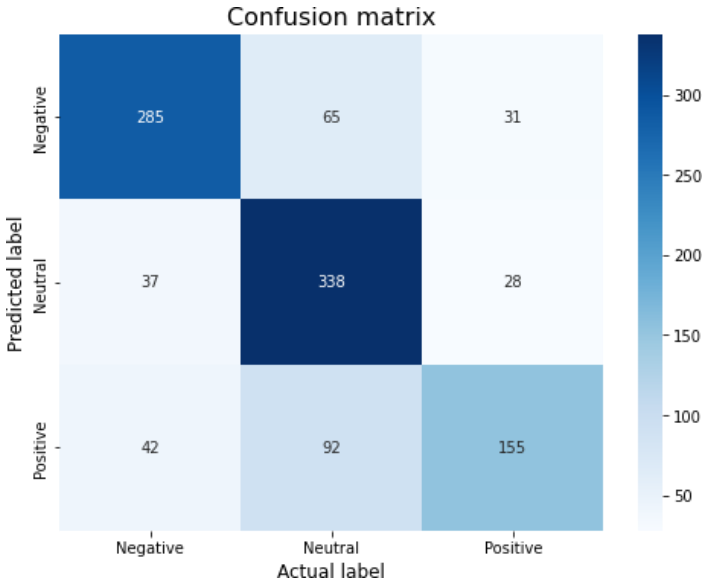

Fig. 15. Training plots for Validation & Training Accuracy

Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks, the SCLE model demonstrated superior predictive performance and interpretability compared to in- dividual CNN and LSTM models, as well as other baseline methods commonly used in stock market prediction. We successfully incorporated a time-series prediction model, a sentiment-analysis model, and a regression model combining sentiments and historic stock values, giving another prediction. Both these predictions were sent to the final meta-model regressor, which is trained on the previous predictions of both models.

Our results demonstrate the potential of Ensemble techniques in Deep Learning for time series prediction. We also demon- strate their ubiquity by employing a similar model architecture for sentiment analysis too. These methods can have practical applications for investment firms, traders, and financial insti- tutions that aim to make informed decisions based on the analysis of large volumes of financial data. The system can also be extended to implement statistical arbitrage. However, that would require extensive computing and knowledge of similar systems, which is beyond the scope of this research.

VI. FUTURE SCOPE

Future research opportunities include the incorporation of more mathematical parameters, macroeconomic indicators, financial reports, and real-time social media sentiment, among other data sources.

Another point worth considering is incorporating attention mechanism into the LSTM Layer, which could further improve model performance by helping focus on relevant features or time steps, allowing the model to adaptively weigh the importance of different input elements.

Different ensemble architectures can be proposed, or be built on top of our proposed architecture.

More modern techniques such as transfer learning, where large pre-trained models such as transformers can be used. They can be fine-tuned to the related domain i.e. stock market prediction. In recent years, transformers have been shown to improve model generalization.

Lastly, our methodology focuses solely on the prediction of the closing value of a stock. It could become a more com- prehensive predictive system by predicting trading volume, macroeconomic indicators, etc.

ACKNOWLEDGMENT

This work has been supported by Fr. Conceicao Rodrigues College of Engineering, Bandra affiliated to the University of Mumbai. The authors express their gratitude towards their mentor Prof. Kranti Wagle for guiding the project. The authors also thank the anonymous reviewers' valuable comments and suggestions for improving the quality of this paper.

REFERENCES

[1] S. Mohan, S. Mullapudi, S. Sammeta, P. Vijayvergia and D. C. Anas- tasiu, "Stock Price Prediction Using News Sentiment Analysis," 2019 IEEE Fifth International Conference on Big Data Computing Service and Applications (BigDataService), Newark, CA, USA, 2019, pp. 205- 208, doi: 10.1109/BigDataService.2019.00035.

[2] Ghosh, Achyut & Bose, Soumik & Maji, Giridhar & Debnath, Narayan & Sen, Soumya. (2019). Stock Price Prediction Using LSTM on Indian Share Market. 10.29007/qgcz.

[3] Hamoudi Hamdy & Elsdifi Mohamed. (2020) Stock Market Prediction using CNN and LSTM.

[4] Nabipour, M., Nayyeri, P., Jabani, H., & Mosavi, A. (2020). Deep learning for Stock Market Prediction.

[5] Mohanty, S., Vijay, A., & Gopakumar, N. (2022). StockBot: Using LSTMs to Predict Stock Prices.

[6] MT Bakker, Forecasting the Stock Market using News Sentiment Analysis (2021)

[7] Sheng Chen and Hongxiang He 2018 IOP Conf. Ser.: Mater. Sci. Eng. 435 012026

[8] Kalyani, J., Bharathi, P. H., & Jyothi, P. R. (2016). Stock trend prediction using news sentiment analysis.

[9] Darapaneni, N., Paduri, A. R., Sharma, H., Manjrekar, M., Hindlekar, N., Bhagat, P., Aiyer, U., & Agarwal, Y. (2022). Stock Price Prediction using Sentiment Analysis and Deep Learning for Indian Markets.

[10] Shi, Z., Hu, Y., Mo, G., & Wu, J. (2022). Attention-based CNN-LSTM and XGBoost hybrid model for stock prediction.

[11] Mehar Vijh, Deeksha Chandola, Vinay Anand Tikkiwal, Arun Kumar, Stock Closing Price Prediction using Machine Learning Techniques, Procedia Computer Science (2020)

[12] Hiransha M, Gopalakrishnan E.A., Vijay Krishna Menon, Soman K.P.,NSE Stock Market Prediction Using Deep-Learning Models, Pro- cedia Computer Science, (2018)

[13] Aditya Bhardwaj, Yogendra Narayan, Vanraj, Pawan, Maitreyee Dutta, Sentiment Analysis for Indian Stock Market Prediction Using Sensex and Nifty, Procedia Computer Science (2015)

[14] Chetan Gondaliya et al 2021 IOP Conf. Ser.: Mater. Sci. Eng. 1020 012023

[15] Lwanga, V. K. (2018). Stock market price prediction using sentiment analysis: a case study of Nairobi stock exchange market (Thesis). Strathmore University

[16] https://github.com/zshicode/Attention-CLX-stock-prediction

[17] https://pypi.org/project/yfinance/

[18] https://technical-analysis-library-in-python.readthedocs.io/en/latest/

[19] Araci, D. (2019). FinBERT: Financial Sentiment Analysis with Pre- trained Language Models.

[20] Kingma, D. P., & Ba, J. (2014). Adam: A Method for Stochastic Optimization.

[21] https://pypi.org/project/nltk/

[22] Django Software Foundation. (2019). Django.