Stock Price Prediction using Generative Adversarial Networks

Fuad Salimzade [1] **Marwin Solomon** [2] **Nikhil Kukreja** [1]

Abstract

Stock price prediction is essential for value investing in the stock market. Predicting stock prices is an extremely notorious task considering the low signal-to-noise ratio present in the data and the ever-evolving markets. But, short-term prediction leveraging financial news has shown promising results in recent years. The difficulty in stock price prediction equipped with the impact that financial news carries on predicting is the two main driving factors behind this paper. Through this paper, we propose a GAN with LSTM-based Generator and Deep-CNN-based Discriminator for predicting stock prices, which considers the effects of financial news obtained in the form of tweets to get a sentiment score, days under consideration to predict the prices and how many days are predicted ahead considering a fixed number of past days. We obtain the sentiment-polarity score of the tweets about different stocks through VADER, a sentiment analysis-based model. Then, the GAN architecture is used to predict the stock prices such that (i) Comparative analysis can be carried out between the baseline models and GAN-based model (ii) Experiments with different parameters such as learning rates, days to consider for predicting stock prices, and the number of days for which closing price is predicted such that optimal parameters can be determined for testing over real-time data. In terms of results, we observe that the GAN model performs considerably well compared to the baseline models and helps in correctly determining the direction of the price one day ahead from the date under consideration for our study.

[1]Department of Mathematics, London School of Economics and Political Science, England [2]Department of Statistics, London School of Economics and Political Science, England. Correspondence to: Fuad Salimzade <f.salimzade@lse.ac.uk>, Marwin Solomon <m.solomon2@lse.ac.uk>, Nikhil Kukreja <n.kukreja@lse.ac.uk>.

1. Introduction

Stock price prediction has always been of interest to people who are actively involved in trading various asset classes such as stocks, commodities, and currencies, among other financial instruments. There have been numerous studies that have attempted to predict it accurately but most of them found it challenging as the amount of signal-to-noise ratio found in the financial data set is fairly less. Moreover, the stock price is also affected by various factors such as interest rates, political and geopolitical events, and market moods. These elements affect investor sentiment and cause changes in stock prices. Hence, it is essential to consider these elements while investing.

Social media has become a significant channel for communication and information sharing, and as a result, it has become a valuable tool for businesses to connect with customers and influence public perception. In the finance industry, social media has become an essential source of data for market analysis and has continuously been seen to have a significant impact on consumer behavior and brand reputation.

In order to efficiently analyze financial data, researchers have found that neural networks work well in predicting the stock price as they are able to capture the hidden essential signals in the financial data to predict it efficiently. With this paper, we aim to explore neural networks to effectively predict stock prices.

Moreover, our paper focuses on addressing the issues faced in the existing literature to predict stock prices by using sentiment analysis to extract information from textual information about stocks to capture the market sentiment. The automated process of understanding an opinion about a given subject from news articles is known as sentiment analysis. It helps to understand if the newspapers are talking positively or negatively about financial markets which help in determining the trend.

In this paper, we first review relevant research literature to understand the landscape and industry. Then, we try to attach a sentiment score to the tweets related to the stocks in question over the date range considered. We use VADER (Valence Aware Dictionary and Sentiment Reasoner) to get the sentiment score. VADER is preferred by the industry as it gives a clear distinction between positive and negative

sentiment. We then try to analyze the stock movement and the effects of the general sentiments for the day by incorporating the sentiment score in the stock prediction model. We leverage the LSTM-based GAN model to predict the stock price. Lastly, we design experiments to compare the different neural network architectures that are recognized by the industry to reveal the interactions between the financial news and the stock movement.

2. Related Work

Stock price prediction has always been crucial for investment firms and quantitative analysts. (Chang et al., 2008) provided insight that predicting stock prices accurately has always been a challenge as it is affected by various factors such as interest rates, inflation rates, and financial news. In order to predict it accurately, attempts have been made to quantify the qualitative data such that it can be utilized in the analysis. However (Li et al., 2019) found that it is still difficult for investors to analyze and predict market trends according to all of this information. (Akita et al., 2016) presented that a lot of artificial intelligence techniques have been investigated to automatically predict those trends. For example, (Devitt & Ahmad, 2007) determined the stock trend by determining the polarity score through a lexicon-based sentiment analysis model.

However, (Gilbert, 2016) found that these works encounter issues with their analysis as they do not consider the proper influence of financial news for stock trend analysis. Earlier studies focused on the Bag-of-Words model for capturing information from textual data that cannot consider any interpretation of linguistic patterns or aspects such as the ordering of words, synonyms, and pronoun resolution.

For integrating financial news into training data, sentiment analysis is one of the methods that is utilized. (Mohan et al., 2019) uses NLTK (Loper & Bird, 2002) to derive a polarity score for their stock prediction task. Hence, to overcome this issue we use NLTK's VADER which was found to be quite successful by (Qin et al., 2017) when dealing with news reviews as it generates the sentiment score according to semantic orientation. VADER, a simple rule-based model was developed by (Gilbert, 2016), who conducted a study then to compare the performance of VADER with eleven other benchmarks and found that VADER outperformed traditional sentiment lexicons like ANEW (Bradley & Lang, 1999) and SentiWordNet (Baccianella et al., 2010) among others.

Furthermore, models such as SVM (Cortes & Vapnik, 1995) and Feed Forward Neural Networks (Rumelhart et al., 1986) are not able to deal with time series data effectively as they are not based on the time-series aspect of stock data. So, we use LSTM-based RNNs shown by (Izumi et al., 2010) which capture the time-series information due to their architecture of memorizing the previous time-steps.

Extensive studies have been conducted in the industry that legitimizes the use of ARIMA (Alonso & Garcia-Martos, 2012) and LSTM-based models for predicting stock movements with low forecast errors and higher accuracy as compared to the traditional models. (Sima Siami-Namini, 2018; Banerjee et al., 2005; Khashei & Bijari, 2011; Alonso & Garcia-Martos, 2012; Ariyo et al., 2014) advocate the benefits of using these models specifically for the economics and financial time-series data with both models providing high accuracy compared to traditional models. Variations and combinations of ARIMA and LSTMs have been constantly researched in the industry for time-series prediction and are the reason these models are used as a baseline to compare the performance of GAN through the course of the paper.

Generative Adversarial Network (GAN) first introduced by (Ian J. Goodfellow, 2014) is a powerful deep learning model that is able to learn the representations of training data and generate similar synthetic data points. Because of their distinct ability to generate synthetic data that are very close to training data distribution, GANs have widely been used in image generation (Reed et al., 2016) and speech generation (Yamamoto et al., 2020) applications such as text-to-image, text-to-video, and text-to-speech. So far, the majority of the work has been done in the field of audio and imaging applications that utilize GANs extensively.

However, we believe that GANs have a predictive power that is surpassing the performance of the discriminative models for time-series prediction. One of the first applications of RNN-based GAN on sequential data has been carried out by (Mogren, 2016). He described how RNNs (Hopfield, 1982) known to be better suited for sequential data are effective when combined within the GAN architecture. In the following years, GAN has been tested specifically on predicting stock prices. In this regard, (Zhang et al., 2019) shows the application of LSTM-based GAN and MLP as a discriminator in predicting the closing price of a stock and how it surpasses the discriminative deep learning models.

Surrounding the preceding work, (Zhou et al., 2018) shows the effectiveness of using GAN which employs a deep LSTM model as a generator and deep CNN as a discriminator to predict the closing price of the stocks. Their model is implemented using a combination of historical data, technical indicators, and textual information from the news. In this paper, our research is based on the application of the architecture developed by (Zhou et al., 2018) to predict the closing price of 10 stocks.

3. Architecture

3.1. GAN Model

3.1.1. Principle

Generative Adversarial Network is a framework that trains two discriminative models at the same time. A high-level overview of the GAN model defined for our problem is in Figure 1. The generator $G(X_0, X_1, ..X_{t-1})$ collects the sequential input data and adds the next generated time-point X_t to the sequential data. Whereas, the discriminator $D(X_0, X_1, ..X_{t-1}, X_t)$ aims to differentiate the incoming sequential data from the fake data.

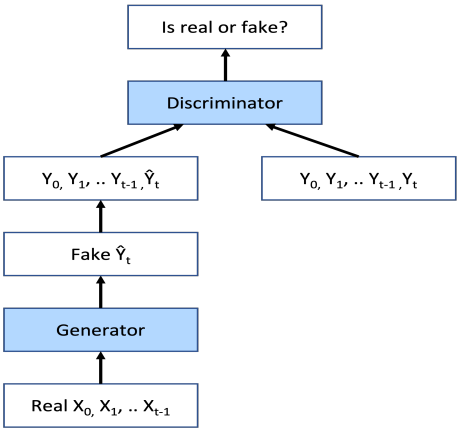

Figure 1: High-level Overview of GAN Architecture

Together, this framework corresponds to a minimax zero-sum game (Ian J. Goodfellow, 2014) where generator $G(\cdot)$ and discriminator $D(\cdot)$ are in competition amongst each other with the generator trying to maximize the probability of the discriminator making a mistake, while the discriminator learning better each time to distinguish a fake data point. A unique solution exists when discriminator $D(\cdot)$ is unable to differentiate the real input sequence from the fake one (Ian J. Goodfellow, 2014). At this cross-section, generator $G(\cdot)$ is able to recover the distribution of training data. Our optimization function value can be defined as follows:

$$\min_G \max_D V(G, D) = \mathbb{E}[\log D(X)] + \mathbb{E}[\log(1 - D(x_0, x_1, ..x_{t-1}, G(z)))]$$

The basic functioning logic behind the GAN is that it is able to generate fake data which makes it impossible even for state-of-the-art deep learning models to distinguish from real data.

In the original paper, (Ian J. Goodfellow, 2014) defines loss function based on the Kullback Leibler - Jensen Shannon divergence, however, in our training process we use cross-entropy to train the GAN model.

3.1.2. Generator

Forecasting the next possible movement of stocks has been of great importance for high-frequency traders and hedge funds in order to base their strategies and perform risk assessments. The prediction model defined in this paper takes into consideration historical price of a stock along with financial indicators, technical information and overall polarity score of a financial news surrounding that stock in a given time t. Our goal is to predict the closing price of a stock at X_{t+1} based on the last 4 consecutive days of stock information.

The architecture defined for generator is based on the LSTM (Hochreiter & Schmidhuber, 1997) model and is shown in Figure 2.

Long Short-Term Memory(LSTM) is a special kind of recurrent neural network that was designed to model long-term dependencies. The basic components of LSTM networks include LSTM layers which are made up of an input gate, an output gate, and a forget gate which together regulates the cell state. The long-term memory is held in the cell state, while the short-term memory is held in the hidden state.

1. Forget Gate: It helps to decide which current and previous information are kept and rejected from the hidden status of the previous run and current status. These values are passed into the sigmoid function. If the value is 0, it means that previous information is forgotten, and when it is 1, it means that previous information is retained.

2. Input Gate: It helps to decide the value of the current input to solve the task. The information essential in the input gate is added to the cell state to form a new cell state used for determining the next cell state.

3. Output Gate: The output of the LSTM model is calculated in Hidden State. The sigmoid function determines the information coming through the output gate, which is multiplied by the cell state after activation with the tanh function.

The input of the generator $G(\mathbf{X})$ is defined as follows:

$$\mathbf{X}_{t,k}^{stock} = [X_{t,h_1}, .., X_{t,h_5}, X_{t,z_1}, .., X_{t,z_6}, X_{t,s}]$$

where t denotes time steps, h denotes historical indicators, z denotes technical indicators and s denotes the polarity score of financial news.

Generator $G(\cdot)$ consists of three LSTM layers with one output dense layer. The LSTM layers have 128, 64, and 32 units respectively. The output of last LSTM layer is fed into feed-forward output layer. The linear activation function is employed in the last layer which provides us

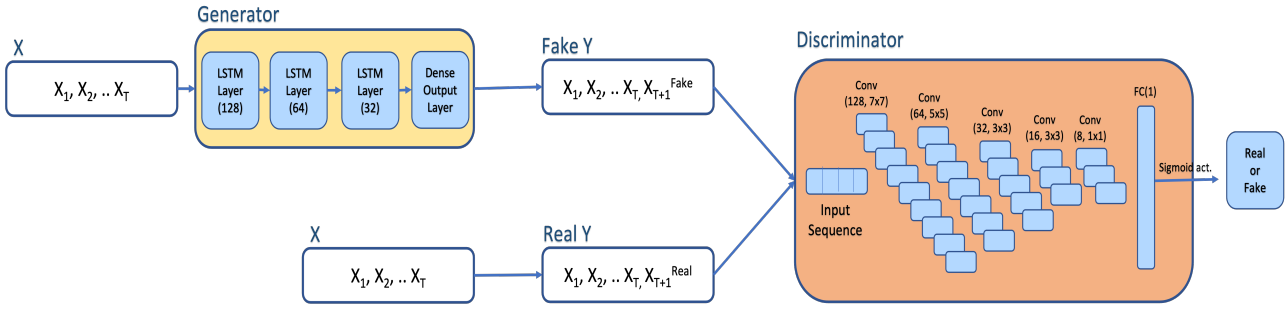

Figure 2: Detailed GAN Architecture

with a scalar value for a closing price. LSTM layers have been widely used for time series prediction thanks to their ability to retain information from previous time steps and to forget information that does not add value.

The output of the generator $G(\mathbf{X})$ is defined as follows:

$$G(\mathbf{X}) = X_{t+1,p}^{stock} = \delta(w * \mathbf{H} + b)$$

where δ denotes a linear activation function in the output dense layer, $\mathbf{H}$ denotes the output of the last LSTM layer, and p denotes the closing price of the stock.

In order to make discriminator $D(\cdot)$ maximum confused so that it will not be able to differentiate the predicted closing price from a real closing price, we define the loss function for $G(\cdot)$ as follows:

$$L_G = L_{bc}(D(G(\mathbf{X})), 1) = L_{bc}(D(\mathbf{X}_{t+1}), 1) \quad (1)$$

where L_{bc} is a binary cross-entropy function defined as:

$$L_{bc} = \frac{1}{m} * \Sigma^m \log(1 - D(\mathbf{X}_{t+1})) \quad (2)$$

(Zhang et al., 2019) and (Zhou et al., 2018) demonstrate the effectiveness of applying LSTM as a prediction model in GAN framework and show how it supersedes GRUs which is another RNN-based model that shares similar architecture designed for sequential data.

3.1.3. Discriminator

The detailed architecture of the discriminator is shown in Figure 2. For discriminator model $D(\cdot)$, we employ Convolutional Neural Networks to discriminate real date from fake data. The model consists of five convolutional layers and one feed-forward output dense layer. Information about the discriminator model is presented in Table 2.

The input to $D(\cdot)$ consists of x days of closing prices concatenated with generated or real closing price in day $t+1$.

$$D(Y_{t-x-1}, .., Y_{t-2}, Y_{t-1}, Y_t, \mathbf{Y}_{t+1}) = \sigma(\cdot) \quad (3)$$

where $\mathbf{Y}_{t+1}$ is either real or generated closing price and σ denotes sigmoid activation function in the output layer of feed forward dense layer.

The output of discriminator model is a scalar value between 0 and 1, which gives the probability that incoming sequential data is from a distribution of real data.

Layers	Parameters
Convolution 1	128, 7x7, stride = 2, LeakyRelu
Convolution 2	64, 5x5, stride = 2, LeakyRelu
Convolution 3	32, 3x3, stride = 2, LeakyRelu
Convolution 4	16, 3x3, stride = 2, LeakyRelu
Convolution 5	8, 1x1, stride = 2, LeakyRelu
Fully Connected	1, Sigmoid

Table 1: Architecture of Disciminator Model

During the training step of the discriminator, we freeze the learnable parameters of generator $G(\cdot)$ in order to retrieve fake output and compare it with real output at each iteration.

Minimizing only generator loss is not enough. This can lead the generator to produce fake outputs that would confuse the discriminator while not approximating the training data distribution. This, in turn, would lead the discriminator to learn the difference, which in turn would lead the generator to try to confuse the discriminator by producing different samples (Zhou et al., 2018). Therefore, it is essential to define the loss function for the discriminator as well. We define loss function for $D(\cdot)$ (Ian J. Goodfellow, 2014) as follows:

$$L_D = L_{bc}(D(G(\mathbf{X})), 0) + L_{bc}(D(\mathbf{X}^{real}), 1) \quad (4)$$

where L_{bc} is a binary cross-entropy function.
The convergence for the GAN framework happens when discriminator $D(\cdot)$ differentiates fake samples by a probability of $\frac{1}{2}$, at which point, generator $G(\cdot)$ recovers the distribution of real data (Ian J. Goodfellow, 2014).

4. Training Methods

For model configuration, we used 500 epochs for training the model. The dataset was split between $85\%/15\%$ for training and test dataset.

4.1. Activation Function

In our discriminator model, we have utilized the Leaky ReLu (Maas & Ng, 2013) activation function. In contrast to the ReLu activation function (Nair & Hinton, 2010), Leaky ReLu is able to handle negative weights with a small slope setting, while ReLu sets weights to zero making information impossible to forward propagate and causing information loss. Negative weight handling has been proven to help with the vanishing gradient problem and outperforms ReLu when used in deep convolutional layers.

In this regard, we have set $\alpha = 0.1$ for Leaky ReLu and Figure 3 shows the difference in handling negative values between ReLu and Leaky ReLu.

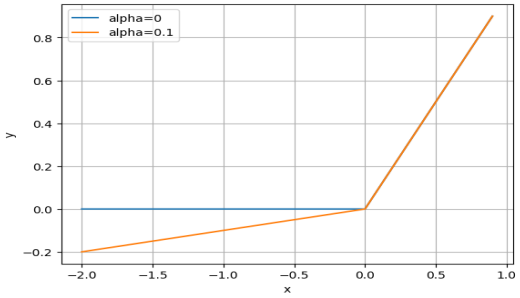

Figure 3: Leaky Relu versus ReLU

4.2. Optimizer Choice

In the GAN framework, we have two models that are in competition with each other. We observed the sudden but frequent volatility in loss function values during the training epochs, which we believe stem from rapid changes in the gradient updates because of generator and discriminator models fighting each other. Adaptive gradient algorithms Adam and AdamW showed better performance in our experimentation.

AdamW proposed by (Loshchilov & Hutter, 2019) is a new variant of Adam (Kingma & Ba, 2014) optimizer. (Loshchilov & Hutter, 2019) mentions that weight decay in Adam, which is known to play the role of L2 regularizer, might be misleading in some circumstances and introduces a modification to Adam. Given the GAN framework, we have adopted AdamW in our implementation. Similar to Adam, we choose $\beta_1 = 0.5$ as the parameter. Choosing $\beta_1 = 0.5$ smooths the momentum to deal with rapidly updated gradients (Zhang et al., 2021). We have experimented with different learning rates and observed learning rate $\rho = 0.001$ yields a better accuracy in our evaluation metrics.

Figure 4 shows the loss value over the epochs. We can see that generator loss stagnates around 0.7, while discriminator loss stagnates around 1.4.

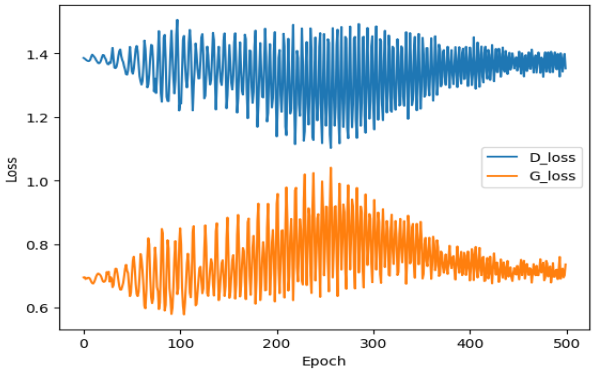

Figure 4: Generator and Discriminator loss over Epochs

4.3. Regularization

To avoid model overfitting, the dropout regularization method has been employed in the generator model with a $rate = 0.3$ to let the model generalize well.

5. Numerical Results

5.1. Goals

Through this paper, we are trying to measure the effect of financial tweets on predicting the stock price by comparing different models. From the existing literature, we know that the task of predicting the stock price is challenging. Hence, to capture the effect of market sentiment we obtain tweets about various stocks and then translate them into quantitative figures through VADER. The sentiment score obtained for each company over the respective dates is leveraged as a feature in our dataset to predict the stock price of various organizations under consideration.

5.2. Dataset

5.2.1. List of Stocks and Time Period

For our dataset, we consider the financial information of the following stocks: 'TSLA', 'TSM', 'AAPL', 'PG', 'AMZN', 'MSFT', 'NIO', 'META', 'AMD', and 'NFLX' over the period of one year from '30-09-2021' to '29-09-2022'.

5.2.2. Data Acquisition and Characterisation

We downloaded the dataset from the 'yfinance' API offered by Yahoo Finance in Python. Our initial dataset downloaded from the 'yfinance' library is characterized by the following

features: 'Open', 'Close', 'High', 'Low', and 'Adj Close' which gives us information about the price of the stock over a day while the 'Volume' feature tells us about the number of shares traded on a given day.

5.2.3. Feature Engineering

Many hedge funds and traders include technical indicators in their stock prediction models as they have been proven to add predictive power to signal noise detection and improve accuracy (Chong et al., 2017; Kim, 2003).

1. Technical Indicators - we add more features by including the technical indicators such as 'MA7', 'MA20', 'EMA', 'MACD', 'ROC', 'Bollinger Bands', and 'Log Momentum'. 'MA7' represents the moving average over 7 days, 'MA20' represents the moving average over 20 days, and 'EMA' represents the exponential moving average. 'MACD' represents the price trend of the stock and stands for moving average convergence divergence, 'ROC' carries information about the momentum of the stock and stands for the rate of change, 'Bollinger Bands' carries information about the volatility of the stock price, and 'Log Momentum' captures information about the momentum of the stock price.
2. Sentiment Score - Furthermore, to capture the market sentiment, we also consider the financial tweets about these companies for the same period. These tweets go through the VADER model to determine the sentiment score for the various companies under consideration.

Figure 5 shows the plots for some of the technical indicators for Apple, Inc. company. Hence, our final dataset is made

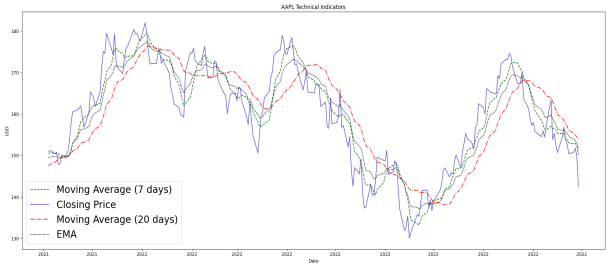

Figure 5: Technical indicators for Apple

up of the following quantitative features which are fed into our model for determining the closing stock price: 'Open', 'High', 'Low', 'Close', 'Volume', 'Adj Close', 'MA7', 'MA20', 'EMA', 'MACD', 'ROC', 'Bollinger Bands', 'Log Momentum', and 'Sentiment Score'.

5.2.4. Feature Transformation

We scale our dataset as proposed by (Sachdeva Akshay & VN, 2019; Guangyu & Liangxi, 2020) such that the different measurement units do not affect our analysis. All the attribute data are normalized to fall within the same range of (0,1). In order to perform scaling, we leverage the min-max normalization functionality.

$$x^{'} = \frac{x - min(x)}{max(x) - min(x)} \tag{5}$$

5.2.5. Data Segmentation

The base dataset, with time series and all the indicators, is divided into rolling segments of 4 days (referred to as M in Figure 6) used for training the model. These 4 days are used to predict the stock price for the 5th (referred to as N) consecutive day in the data set. The rolling segmentation (Zhou et al., 2018) approach for stock prediction is a standard practice used in the industry that is used throughout the course of the paper. Before training the final model, several experiments were conducted to choose the optimal values of M and N. In regards to the experimentation, 4 days have been chosen for M and it is also indicative of the stock trading frequencies practically applicable for the traders.

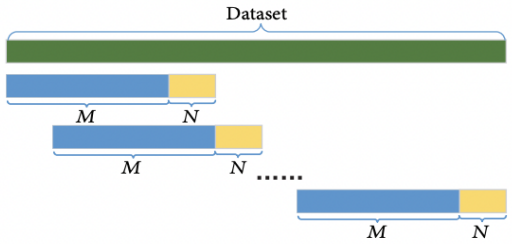

Figure 6: Train/Test Data Split

5.3. Evaluation Metrics

We consider our primary evaluation metrics as 'Root Mean Squared Error' (RMSE) for determining how well we predict the adjusted closing price for the stocks under consideration.

We define root mean squared as follows:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{6}$$

where n is the number of data points under consideration, y_i is the actual value of the i^{th} data point, and $\hat{y}_i$ is the predicted value of the i^{th} data point which is obtained from our model.

We choose RMSE as our evaluation metric similar to works by (Zhou et al., 2019; Sima Siami-Namini, 2018) , as it is sensitive to errors in the predictions and has the same unit as the quantity that is being predicted through the model. Moreover, it is differentiable in nature and hence useful for

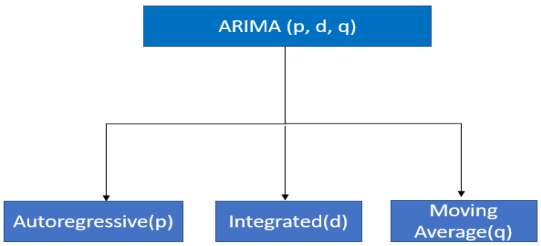

Figure 7: ARIMA Model

calculating optimal parameters in the model when optimizing the loss function to achieve minimum loss.

Smaller values of root mean squared error mean that our prediction is very close to the original adjusted closing price and our model generalizes well.

5.4. Baseline Models

For baseline models, we trained ARIMA and two different models using LSTMs, a shallow LSTM network, and a deep LSTM network. These models are basic and widely used variations of the RNN models and are known for their simplicity, versatility, and interpretability in the industry.

5.4.1. ARIMA Model

ARIMA as discussed by (Alonso & Garcia-Martos, 2012) and shown in Figure 7 stands for Autoregressive Integrated Moving Average, is a classical model used for forecasting the values of a time series and is widely used as a benchmark model for comparison purposes. This type of model is essential as it captures dependency among observations by incorporating information from past observations along with trends and seasonal patterns.

The three main components of the ARIMA model are as follows:

1. Autoregression (AR) - This component represents the relationship between an observation and a lagged value of itself. Thus, it helps in modeling the dependency of a variable with its past values.
2. Moving Average (MA) - This component represents the relationship between an observation and a residual error from a moving average model applied to lagged values of the series. Thus, it helps in modeling the dependency of a variable with its past errors.
3. Integration (I) - This component represents the number of times differencing is required to achieve the stationarity, i.e., a constant mean and variance over time.

We choose the ARIMA model as a benchmark in our study as it can capture a wide range of patterns including trends, seasonality, and cyclical relationships. Moreover, these models can be easily interpreted and the parameters of the model can be used to identify the strength and nature of the relationship between the variables.

5.4.2. Shallow LSTM

The Shallow LSTM network is a two-layer network that takes in the sequential time-series data to give out a prediction based on the training. It utilizes LSTM as the first layer, consisting of 64 neurons. The output of the LSTM layer is then flattened and passed onto an output Dense layer.

For the Shallow LSTM network, we use ADAM as the optimizer with a learning rate of 0.001. The preferred choice of activation function for the output layer of the shallow LSTM network is the sigmoid function. We use an L1 regularizer for weight regularization of our shallow LSTM network. We complement the L1 regularizer to prevent overfitting by adding a Dropout layer.

Shallow LSTM is the most basic LSTM neural network that is known in the industry to work well for small data sets and detect simple pattern, and is shown to perform better than Deep LSTM models occasionally (Bhandari et al., 2022). Its ease of result interpretation and experimentation along with its computational efficiency and high accuracy, are the reasons of having a shallow LSTM network as a baseline for comparison in the analysis.

5.4.3. Deep LSTM

The Deep LSTM network utilizes multiple LSTM and Dense layers in the architecture to generate the results. The model has 2 LSTM layers with 32 and 16 as the number of neuron units respectively. The LSTM layers are followed by two Dense layers with 16 and 8 units respectively. The output of the last dense layer is then flattened and passed to the output Dense layer.

Similar to the Shallow LSTM network, we use ADAM as the optimizer for our Deep LSTM network. We use multiple activation functions for the Deep LSTM network. For the intermediate dense layers, we use tanh as the activation function while for the final dense layer, we use sigmoid as the activation function. Similar to the shallow LSTM network, we use an L1 regularizer for the regularisation of weights in our Deep LSTM network.

Deep LSTM networks have proven to be powerful tools when dealing with complex time-series data with long-term dependencies and have shown to be robust against noise. (Althelaya et al., 2018) showed how deep or stacked LSTM networks outperform shallow LSTM for stock movement prediction for both long-term forecastings. The similarities of the Deep LSTM architecture with the Generator model used in the GAN architecture make it an interesting choice to add as a baseline model in the comparison analysis of the

paper.

5.5. Experimentation

Several experiments are conducted to understand the effect of parameters on the GAN model while training. These experiments were restricted to one specific stock 'AMZN' and generalized based on the domain of the stocks in consideration.

Table 2 shows the GAN model that was trained by considering different windows of days to predict the next closing day. Intuitively, the lesser the number of days in the window should make the model perform better. The results although didn't confirm any such pattern.

Days to Consider	3	4	5	7
RMSE	6.84	6.79	5.79	7.16

Table 2: Experiments with Days to consider for prediction

Table 3 shows the GAN model that was trained to predict the stock price for different days (2 and 3) based on a window of 4 days. This attests that the GAN model is more suitable for 1-day short-term forecasting and has a limitation when it comes to long-term forecasting.

Days to Predict	2	3
RMSE	10.5	14.42

Table 3: Experiments with Days to predict

Experimenting with learning rates, using 4 days to predict the stock price of 1 day, we observed 1e-3 provides the optimal result.

Learning Rates	1e-3	1e-4	1e-5
RMSE	5.79	8.59	8.65

Table 4: Experiments with Learning rates

Our model has been trained on a dataset within the time frame of 2021 and 2022. In addition to experimentation, we utilized Twitter API and tested our model on the date this paper was written. Over 8000 tweets from '20-04-2023' to '25-04-2023 have been extracted about 'AMZN' stock along with corresponding stock price information and it was evaluated using the pre-trained GAN model to predict stock price on '26-04-2023'. The resulting RMSE is 0.045.

Stock	Real	Predicted
AMZN	104.98	104.94

Table 5: Predicting stock price on '26-04-2023'

5.6. Results

The final model was tested on the test data set for all ten stocks in consideration for the analysis. Intuitively, the GAN model is bound to show different results for different stocks when compared to the baseline models because of different scales and the fact that the best model hyperparameters were selected by training for one particular stock 'AMZN'. The GAN model provided better results than the ARIMA model for all ten of the stocks in consideration but performed better for 80% of the stocks when compared to the shallow-LSTM model. The deep-LSTM network, due to its similarities in architectures with the Generator architecture of the GAN model (both using a combination of LSTM) was expected to show good results. Despite the similarities, the GAN model performed better for 60% of the stocks in consideration. All the individual models (GAN and Baselines) were optimized to give out the best results and test the performance in real-life situations.

Stock Name / RMSE	GAN	ARIMA	Shallow-LSTM	Deep-LSTM
TSLA	8.9	19.9	12.2	9.7
TSM	2.7	7.6	3.8	3.2
AAPL	4.7	9	4	3.7
PG	4.2	6	3.1	2.3
AMZN	3.5	14.7	4	5.5
MSFT	9.3	25.8	10.1	6.3
NIO	1.2	2	1.5	1
META	8.8	16.4	12.5	12.3
AMD	4.2	21.2	5.1	4.4
NFLX	9.6	9.3	11.3	13.2

Table 6: Stock RMSE comparison

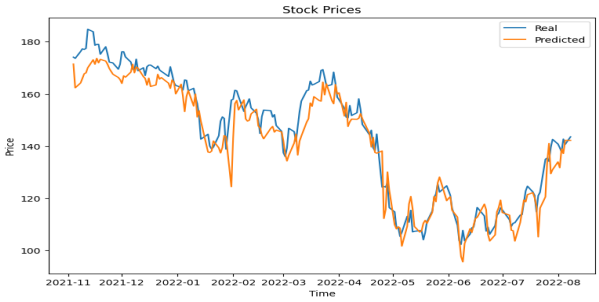

Figure 8: Generalization of GAN Model Over Training data of 'AMZN' Stock

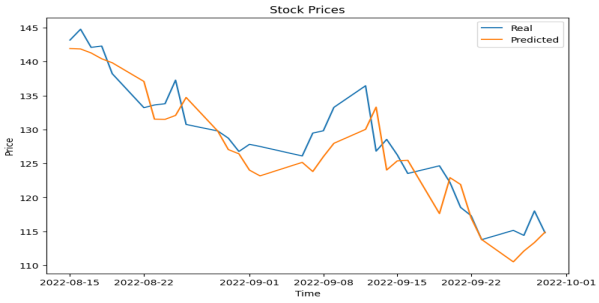

Figure 9: GAN Model Evaluation on Test Data of 'AMZN' Stock

6. Conclusion

The paper helped understand the impact advanced deep learning algorithms can have to understand the complex and

noisy patterns in the stock markets. A GAN model, comprising an LSTM-based Generator and Deep-CNN-based Discriminator framework was proposed and a comparative analysis confirmed a better performance for at least 60% of the stocks in consideration than the traditionally recognized baseline models. The model was ultimately tested on a real-time dataset obtained for the period between 20-04-2023 and 26-04-2023 to test the legitimacy of the model in keeping up with the ever-evolving markets. Public sentiments, which are highly based on the financial news will always be a key component in improving the accuracy of the prediction. In regards to long-term forecasting, our model has been shown to have low accuracy for predicting the next 2 and 3 days, therefore, the proposed GAN model can be valuable over a 1-day short-term forecasting horizon. It'll be interesting to extend the analysis further to see the adaptation of the model as a trading strategy in the real world.

Statement about individual contributions

Fuad Salimzade: Contributed to the research of the project, designed and presented the GAN model, along with the experimentation and building of a data extraction pipeline for Twitter API utilization.

Marwin Solomon: Contributed to the research of the project, designed and presented the baseline ARIMA models, data pre-processing along with the result interpretation and optimization.

Nikhil Kukreja: Contributed to the research of the project, designed and presented the baseline LSTM models, along with the result interpretation and optimization.

References

Akita, R., Yoshihara, A., Matsubara, T., and Uehara, K. Deep learning for stock prediction using numerical and textual information. In *2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS)*, pp. 1–6. IEEE, 2016.

Alonso, A. and Garcia-Martos, C. Time series analysis-forecasting with arima models. *Universidad Carlos III de Madrid, Universidad Politecnica de Madrid*, 2012.

Althelaya, K. A., El-Alfy, E.-S. M., and Mohammed, S. Stock market forecast using multivariate analysis with bidirectional and stacked (lstm, gru). In *2018 21st Saudi Computer Society National Computer Conference (NCC)*, pp. 1–7, 2018. doi: 10.1109/NCG.2018.8593076.

Ariyo, A. A., Adewumi, A. O., and Ayo, C. K. Stock price prediction using the arima model. In *2014 UKSim-AMSS 16th international conference on computer modelling and simulation*, pp. 106–112. IEEE, 2014.

Baccianella, S., Esuli, A., Sebastiani, F., et al. Sentiwordnet 3.0: an enhanced lexical resource for sentiment analysis and opinion mining. In *Lrec*, volume 10, pp. 2200–2204, 2010.

Banerjee, A., Marcellino, M. G., and Masten, I. Forecasting macroeconomic variables for the new member states of the european union. *Available at SSRN 711161*, 2005.

Bhandari, H. N., Rimal, B., Pokhrel, N. R., Rimal, R., Dahal, K. R., and Khatri, R. K. Predicting stock market index using lstm. *Machine Learning with Applications*, 9:100320, 2022. ISSN 2666-8270. doi: https://doi.org/10.1016/j.mlwa.2022.100320. URL https://www.sciencedirect.com/science/article/pii/S2666827022000378.

Bradley, M. M. and Lang, P. J. Affective norms for english words (anew): Instruction manual and affective ratings. Technical report, Technical report C-1, the center for research in psychophysiology ..., 1999.

Chang, P.-C., Fan, C.-Y., and Liu, C.-H. Integrating a piecewise linear representation method and a neural network model for stock trading points prediction. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 39(1):80–92, 2008.

Chong, E., Han, C., and Park, F. C. Deep learning networks for stock market analysis and prediction: Methodology, data representations, and case studies. *Expert Systems with Applications*, 83:187–205, 2017.

Cortes, C. and Vapnik, V. Support-vector networks. *Machine learning*, 20:273–297, 1995.

Devitt, A. and Ahmad, K. Sentiment polarity identification in financial news: A cohesion-based approach. In *Proceedings of the 45th annual meeting of the association of computational linguistics*, pp. 984–991, 2007.

Gilbert, C. J. H. E. Vader: A parsimonious rule-based model for sentiment analysis of social media text. 2016.

Guangyu, D. and Liangxi, Q. Study on the prediction of stock price based on the associated network model of lstm. *International Journal of Machine Learning and Cybernetics*, 11:1307–1317, 2020.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. 1997.

Hopfield, J. J. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.

Ian J. Goodfellow, Jean Pouget-Abadie, M. M. B. X. D. W.-F. S. O. A. C. Y. B. Generative adversarial networks. 2014.

Izumi, K., Goto, T., and Matsui, T. Trading tests of long-term market forecast by text mining. In *2010 IEEE International Conference on Data Mining Workshops*, pp. 935–942. IEEE, 2010.

Khashei, M. and Bijari, M. A novel hybridization of artificial neural networks and arima models for time series forecasting. *Applied soft computing*, 11(2):2664–2675, 2011.

Kim, K.-j. Financial time series forecasting using support vector machines. *Neurocomputing*, 55(1-2):307–319, 2003.

Kingma, D. and Ba, J. Adam: A method for stochastic optimization. 2014.

Li, X., Li, Y., Zhan, Y., and Liu, X.-Y. Optimistic bull or pessimistic bear: Adaptive deep reinforcement learning for stock portfolio allocation. *arXiv preprint arXiv:1907.01503*, 2019.

Loper, E. and Bird, S. Nltk: The natural language toolkit. *arXiv preprint cs/0205028*, 2002.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. 2019.

Maas, Andrew L, H. A. Y. and Ng, A. Y. Rectifier nonlinearities improve neural network acoustic models. 2013.

Mogren, O. C-rnn-gan: Continuous recurrent neural networks with adversarial training. *arXiv:1611.09904v1*, 2016.

Mohan, S., Mullapudi, S., Sammeta, S., Vijayvergia, P., and Anastasiu, D. C. Stock price prediction using news sentiment analysis. In *2019 IEEE fifth international conference on big data computing service and applications (BigDataService)*, pp. 205–208. IEEE, 2019.

Nair, V. and Hinton, G. E. Rectified linear units improve restricted boltzmann machines. 2010.

Qin, Y., Song, D., Chen, H., Cheng, W., Jiang, G., and Cottrell, G. A dual-stage attention-based recurrent neural network for time series prediction. *arXiv preprint arXiv:1704.02971*, 2017.

Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., and Lee, H. Generative adversarial text to image synthesis. In *International conference on machine learning*, pp. 1060–1069. PMLR, 2016.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

Sachdeva Akshay, Jethwani Geet, M. C. B. M. K. and VN, A. An effective time series analysis for equity market prediction using deep learning model. In *2019 International Conference on Data Science and Communication (IconDSC)*, pp. 1–5. IEEE, 2019.

Sima Siami-Namini, A. S. N. Forecasting economics and financial time series: Arima vs. lstm. 2018.

Yamamoto, R., Song, E., and Kim, J.-M. Parallel wavegan: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6199–6203, 2020. doi: 10.1109/ICASSP40776.2020.9053795.

Zhang, A., Lipton, Z. C., Li, M., and Smola, A. J. Dive into deep learning. *arXiv preprint arXiv:2106.11342*, 2021.

Zhang, K., Zhong, G., Dong, J., Wang, S., and Wang, Y. Stock market prediction based on generative adversarial network. *Procedia computer science*, 147:400–406, 2019.

Zhou, F., Zhou, H.-m., Yang, Z., and Yang, L. Emd2fnn: A strategy combining empirical mode decomposition and factorization machine based neural network for stock market trend prediction. *Expert Systems with Applications*, 115:136–151, 2019.

Zhou, X., Pan, Z., Hu, G., Tang, S., and Zhao, C. Stock market prediction on high-frequency data using generative adversarial nets. *Mathematical Problems in Engineering*, 2018.