DEEP LEARNING BASED STOCK PRICE PREDICTION

ABIWAQAS Y
COMPUTER SCEINCE ENGINEERING BANNARI AMMAN INSTITUE OF TECHNOLOGY

VENKATESH V
COMPUTER SCEINCE ENGINEERING BANNARI AMMAN INSTITUE OF TECHNOLOGY

VISHWA ATHIYAMAN K S
COMPUTER SCEINCE ENGINEERING BANNARI AMMAN INSTITUE OF TECHNOLOGY

ABSTRACT:

Deep learning has lately achieved remarkable performance in a range of industries due to its significant data processing capabilities. Stock market forecasting, portfolio optimization, financial data processing, and trade execution strategies are just a few examples of where it's been used. Stock market forecasting is one of the most popular and indispensable fields in finance. Existing methods for applying generative adversarial networks (GANs) to sequential data do not account for the temporal correlations that are specific to time-series data. Simultaneously, supervised models for sequence prediction—which allow for more precise control of network dynamics and are fundamentally deterministic—are being developed. We offer a new framework for creating realistic time-series data that combines the freedom of unsupervised training with the control of supervised training. The Recurrent Neural Networks Module the Long Short-Term Memory (LSTM) as the discriminator and generator is used in this paper to suggest a new Generative Adversarial Network (GAN) architecture for forecasting stock closing prices. LSTM designed generator to mine stock data distributions from supplied data in the stock market and generate data in the same distributions and discriminator to distinguish between existing time series stock data and generated data. When compared to previous machine learning and deep learning models, experimental results suggest that our GAN can get a promising performance in closing price prediction on real data.

Keywords*: *Generative Adversarial Networks, Stock Market Price, Regression, Deep Learning, Neural Networks

I. Introduction:

The Generative Adversarial Network (GAN) is a deep neural network design that generates bogus data from a feature vector z which is randomly sampled. An unsupervised adversarial learning technique is used in this architecture to learn the pattern probability distribution p(z) of the real data X in the training dataset. The generator G and discriminator D networks are in overall control of this mechanism. The discriminator trains how to distinguish actual data from fake data by enabling the generator synthesize fake data from a specified feature vector.

By receiving the data from the discriminator, the generator evolves to provide more precise information as the discriminator tries to discriminate them. Producing better bogus data complicates the differentiation challenge, and the discriminator improves their ability to distinguish between the fakes and the reals. Adversarial learning is the term coined to the minimax team that played by both networks. The discriminator seeks to maximize the squared error, whereas the generator strives to diminish it. Equation 1 illustrates the Binary Cross-Entropy (BCE) error function used in training GAN models. When the discriminator can no longer distinguish between actual and bogus data, the learning is shown to be accomplished. Figure 1 depicts the GAN architecture.

Since the GAN's introduction, numerous studies have been conducted in various fields to utilize this network design in practical systems such as fabricating synthetic images, generating authentic time series data, producing tune from song lyrics, and enhancing training data. The most concentrated study areas for GANs in finance are creating realistic time series data and forecasting index prices. Several GAN models are built and trained on daily stock price of a stock market index in this research to forecast the price of the subsequent day when the previous week's data is given. The efficacy of the built models is evaluated, and the results are illustrated.

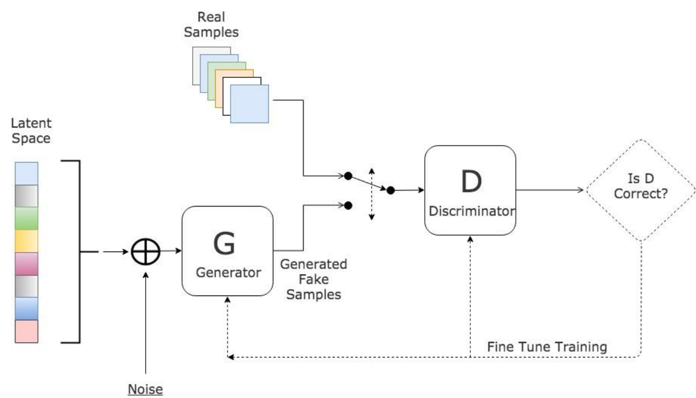


Fig.1: GAN Structure

Retrieved from :https://www.kdnuggets.com/wp-content/uploads/ generative-adversarial-network.png

$$L_{BCE} = \sum_{x^t \in X} \log(D(x^t)) + \sum_{z^t \in p(z)} \log(1 - D(G(z^t))) \quad (1)$$

II. METHODOLOGY

The goal of this research was to see how using the GAN architecture for stock market price forecasting skewed the accuracy. The BIST 100 (XU100) index is used for both training and testing the models' predictive abilities. The daily historical data is collected from the source supplied by Yahoo Finance and comprises the years 01-01-2000 to 01-12-2019. The following are the main features that are employed in training:

- Close Price: The index's most recent traded price
- Open Price: The index's baseline price at the start.
- Highest Price: The index's top price in a single day.
- Lowest Price: The index's cheapest price in a single day.
- RSI 14: Relative Strength Index of two weeks
- SMA 5: Simple Moving Average of 5 days
- CUMLOGRET 1: A single day's compounded log return
- MACD (12, 26, 9): Average of Moving Averages Convergence Divergence for 12 (2 weeks), 26 (1 month) and 9 (1.5 week)

The first four features are collected straight from the dataset, while the other technical categories of data are calculated using the existing data. Following the acquisition of historical data for the eight features, all rows with NaN values are eliminated and the values are normalized using the Min-Max Scaler. After the data preprocessing, 4252 rows remained. Finally, the data is divided into 7-day overlapped segments. The first 70% of the dataset is designated as the training data, while the remainder is divided into six halves (20% -10%) for validation and test sets, respectively.

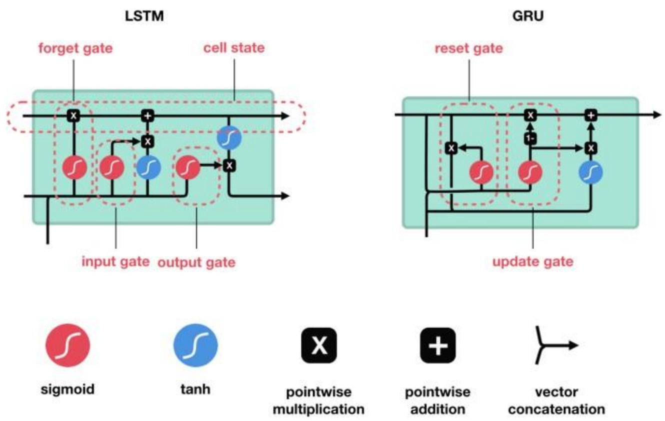


Fig.2: LSTM vs GRU

The PyTorch framework is used to develop one simple regressor and three distinct GAN models with minor improvements for the experimental tests. The simple regressor model is used as a baseline against which the three distinct GAN models are evaluated. The generator entities in these GAN models were kept comparable, and the discriminator components were developed using artificial neural networks (ANN), long short-term memory (LSTM), and gated recurrent units (GRU) architectures. The ANN is a simplistic neural network, but the LSTM and GRU are recurrent neural network (RNN) structures that are primarily employed to process data sequences. The gate ratios inside the divisions and their operating principles change between LSTM and GRU, as indicated in Figure2·

Rather than using the collected z values from the Gaussian distribution as subjected in predefined GAN models, the generator uses LSTM model to estimate the selected features of the next day from the provided statistical data of the previous six days. The learning rate, size of hidden layer, number of epoch and sequence length (number of dates and times used for training set sequences) were customized for the hyperparameter optimization stage. The Root Mean Squared Error (RMSE) is adopted as an error metric (see Equation2) and computed for each attribute in the validation data during the training period in order to meticulously equate and analyze the models.

$$RMSE = \sqrt{\frac{1}{n}\sum_{t=1}^{t=n}(y' - y)^2} \qquad (2)$$

The finest evaluating models are determined by the validation dataset's minimum closing price RMSE values at the end of the evaluation procedure. The model's projections on the testing set were then visualized using the Matplotlib tool, and the RMSE values were provided to indicate the generalization ability.

2.1 EXPERIMENTAL SETUP

The models described in the Methodology part are deployed and their hyperparameters are tweaked for the experiments. The stochastic gradient descent optimization technique Adam is employed, and the number of iterations is defined to 1. The effectiveness of the trained magnitude of the historical data and the batch size are set to 7 and 64, respectively, for each 350 epochs. The learning patterns of the GAN models are assessed using the validation dataset. The order of events For GAN models, the learning rate is set to 5.25×10^{-4}, however for the basic regressor baseline, it is set to 5.12×10^{-4}. Here both generator and the discriminator were maintained the same. The next parts go into the design features of the models that were built.

2.3 SIMPLE LSTM REGRESSOR

For the simple regressor, an LSTM network is employed, and the model is used as a standard against which GAN models can be assessed. During the training, historical data from the previous six days is given into the model, and the model outputs possible trends for the next day. During the training the model, the Mean Squared Error (see Equation 3) between both the actual data of the target and the projected values is backpropagated. Figure 3 depicts this structure.

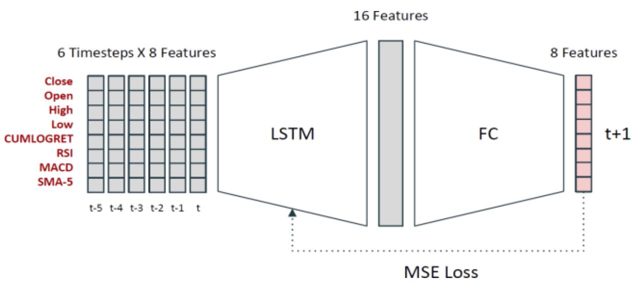


Fig.3: Simple LSTM Regressor Structure

$$MSE = \frac{1}{n}\sum_{t=1}^{t=n}(y' - y)^2 \qquad (3)$$

2.4 GAN WITH ANN DISCRIMINATOR

The generator network of the GAN architecture utilizes the setup of the Simple LSTM Regressor network. The discriminator network is an artificial neural network (ANN) that takes a single vector of 7-day concatenated feature values. The estimated feature vector is combined with the actual feature vectors from the previous 6 days for the fake data, and the resulting vector is labelled as fake. The real labels, on the other hand, are the combined feature vectors from the training dataset for the 7 days. The discriminator learns how to distinguish between actual and fake data. The training is done with the BCE loss function. Figure 1 illustrates the model's structure.

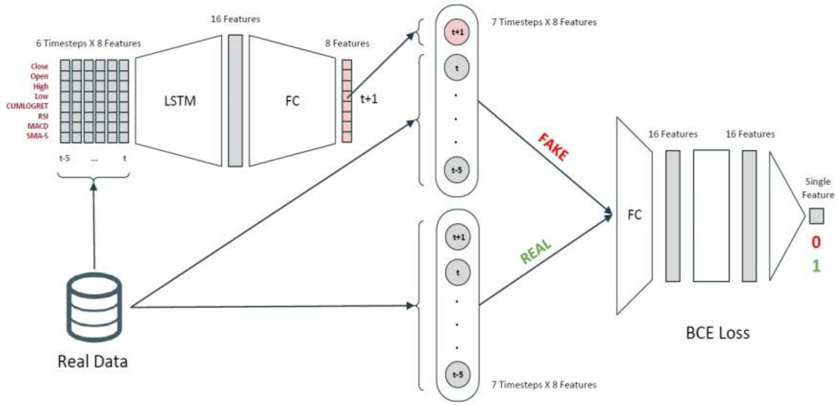


Fig.4: GAN with ANN Discriminator Structure

2.5 GAN WITH LSTM DISCRIMINATOR

The discriminator is converted to an LSTM network, while the generator is kept identical to the GAN with ANN model. Instead of constructing a single vector by combining the feature vectors of the 7 days, the feature vectors are given as sequential data because the LSTM network requires sequential data as input. Figure 1 shows how to build up this model. 5

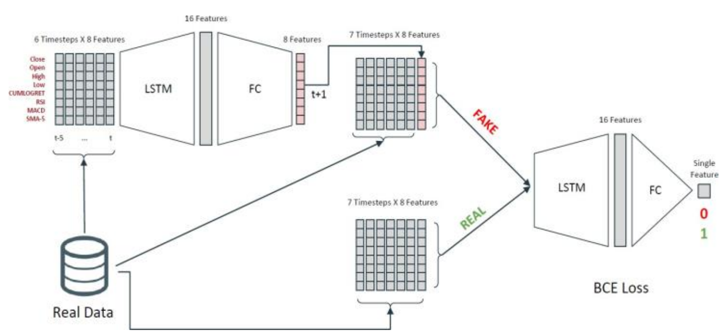


Fig.5: GAN with LSTM Discriminator Structure

2.6 GAN WITH GRU DISCRIMINATOR

With the exception of the GRU being used instead of the LSTM network for the discriminator, this mod is very similar to the GAN with an LSTM discriminator. Figure 6 depicts the structure.

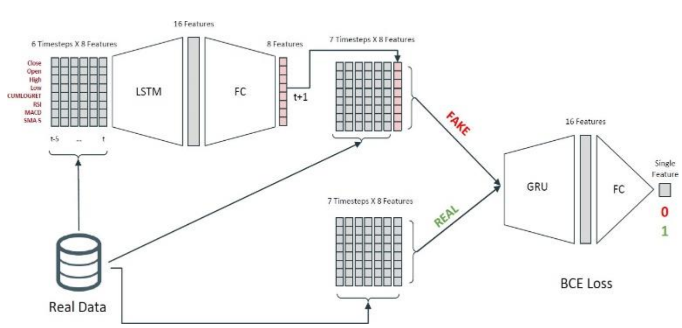


Fig.6: GAN with GRU Discriminator Structure

III. RESULTS AND DISCUSSION

The highest performing weights for every models are selected based on the near price RMSE findings for the validation dataset. Table I lists the epoch numbers of the models that achieved the best, as well as their price RMSE scores on the validation and test datasets.

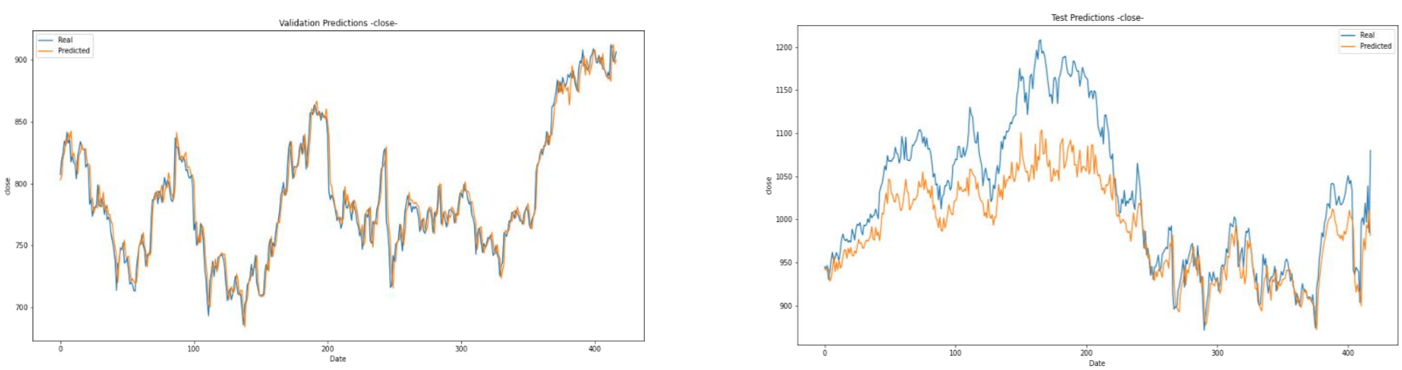
	Epoch	RMSE Validation	RMSE Test
Simple LSTM Regressor	37500	10.8510	50.2491
GAN with ANN Disc.	13500	15.8889	64.3936
GAN with LSTM Disc.	98000	13.9889	31.4466
GAN with GRU Disc.	12000	18.9767	66.5857

TABLE I: Overall performance results of the models

When all of the models' performance on the validation dataset was assessed, the simple LSTM regressor, or the baseline, did better than expected. When the results of the closing price RMSE on the test dataset were considered, however, they were actually worse than the validation errors. Figure 7 shows the model's prediction results on the validation and test datasets. It may well be inferred that, despite the hyperparameters being tweaked, the models still are overfitted because they failed to properly predict close price values in the test dataset that were greater than the training set's values. In Appendix A, you'll find the RMSE findings (see Figure A.1) as well as the prediction results (see Figure A.2).

The ANN discriminator was used in the GAN model which exceeded the basic model. Although being placed third in terms of validation dataset performance, the model dominated the competition on the test dataset by a wide margin. Figure 8 shows that, rather than overfitting the training dataset as the baseline mode, the model was able to generalize the regression model.

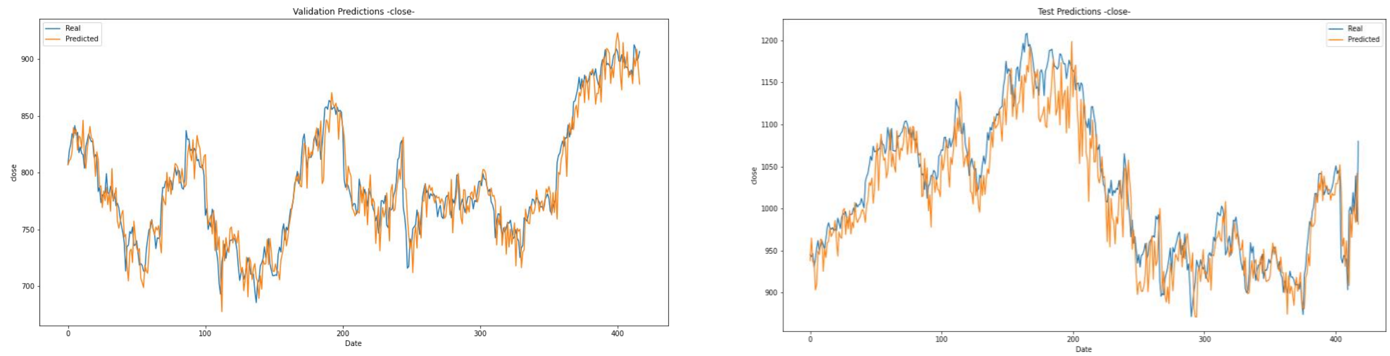
On the test dataset, the GAN models using discriminator with RNN networks that are LSTM and GRU failed on RMSE results. Appendices B, C, and D, respectively, include the final results of the GAN models using ANN, LSTM, and GRU.



(a) Validation Data

(b) Test Data

Fig. 7: Prediction performance of the simple LSTM Regressor



(a) Validation Data (b) Test Data

Fig. 7: Prediction performance of the LSTM Discriminator

Because of the adversarial learning algorithm, the GANs are very sensitive to even the smallest changes in the hyperparameters, so modifying them might provide space for improvement. The results show that employing the ANN network improved the GAN models' performance. Further implication is that utilizing more sophisticated models, which may be achieved by increasing layer sizes, might lead to overfitting. Despite the fact that the validation dataset's price range is closer to the training dataset's values, such overfitted models have clearly failed on the validation dataset.

IV. CONCLUSION:

The GAN architecture could be used to predict asset prices, as observed in the present. GAN's might very well possibly make forecasts on any data in the form of time series, according to this assertion. The results of this study suggest that building a basic LSTM regressor, or base model, were not doing well on past data for the XU100 index. It could figure out how the feature values in the training data relate to one another. When the values vary in the future (validation) data are extended, however, the predictions cannot exceed the value ranges on the training dataset.

As a result, the model is determined to be overfitted. Numerous GAN models using various types of discriminator networks, such as ANN, LSTM, and GRU, are examined by retaining the generator architecture constant. Spite of the fact that the basic model exceeded all other models on the validation dataset due to near price RMSE values, the GAN model with ANN discriminator greatly exceeded the base model on the test dataset. The greater value ranges in the test dataset account for the effectiveness disparity.

Hence, the GAN model could adapt to interpret the data more effectively than that of the LSTM regressor. The other GAN models, which included LSTM and GRU discriminators, were unable to determine the price data. Because GAN models are hyperparameter-sensitive owing to adversarial learning, fine-tuning the characteristics may enhance performance. However, that will not be the case in the examinations. Finally, the trials revealed that due to the overfitting issue, utilizing more sophisticated networks for the generator or discriminator models contributed to higher close price RMSE values.

V. REFERENCES:

[1]. Ian J. Goodfellow, Jean Pouget-Abadie, MehdiMirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio (2014). Generative Adversarial Networks. Retrieved fromhttps://arxiv.org/pdf/ 1406.2661.pdf

[2]. J. Patel, S. Shah, P. Thakkar, and K. Kotecha (2015), "Predicting stock and stock price index movement using Trend Deterministic Data Preparation and machine learning techniques
Al Gharakhanian (2017). Generative Adversarial Networks – Hot Topic in Machine Learning. Retrieved fromhttps://www.kdnuggets.com/2017/ 01/generative-adversarial-networks-hot-topic-machine-learning.html

[3]. Alec Radford, Luke Metz, Soumith Chintala (2016). Unsupervised Rep- resentation Learning with Deep Convolutional Generative Adversarial Networks. Retrieved fromhttps://arxiv.org/pdf/1511.06434.pdf

[4]. Crist ́obal Esteban, Stephanie L. Hyland, Gunnar Ra ̈tsch (2017). Real- valued (Medical) Time Series Generation with Recurrent Conditional GANs. Retrieved fromhttps://arxiv.org/pdf/1706.02633.pdf

[5]. Martin Arjovsky, Léon Bottou - Principled Methods for Training Generative Adversarial Networks

[6]. Takeru Miyato, Toshiki Kataoka, Masanori Koyama, Yuichi Yoshida - Spectral Normalization for Generative Adversarial Networks

[7]. John Cristian Borges Gamboa Deep Learning for Time-Series Analysis

[8]. Torres, J.F. Galicia, A. Troncoso, A. Martínez-Álvarez A scalable approach based on deep learning for big data time series forecasting

[9]. K. Zhang, G. Zhong, J. Dong, S. Wang, and Y. Wang (2019), "Stock Market Prediction Based on Generative Adversarial Network,"

[10]. A. Borovykh, S. Bohte, and C. W. Oosterlee (2017), "Conditional time series forecasting with convolutional neural networks,"