Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 9, Number 1, January 2025 e-ISSN : 2541-2019
DOI : https://doi.org/10.33395/sinkron.v9i1.14246 p-ISSN : 2541-044X

Stock Price Prediction Using TCN-GAN Hybrid Model

Lim Yong Teck[1)*], Angelina Pramana Thenata[2)]
[1,2)]Universitas Bunda Mulia, Tangerang, Indonesia
[1)]aateck.2002@gmail.com, [2)] angelina.pramana@outlook.com

Submitted :Nov 22, 2024 | **Accepted** : Dec 15, 2024 | **Published** : Jan 8, 2025

Abstract: The stock market plays a vital role in national economies, offering significant profit opportunities for investors while exposing them to substantial risks due to market uncertainties. Stock prices often experience significant fluctuations, making accurate prediction a challenging task. Temporal Convolutional Network (TCN) and Generative Adversarial Network (GAN) are the deep learning method proposed for this research. The purpose of this research is to analyze how well the TCN-GAN model predicts stock prices. Previous researches show both TCN and GAN perform well on time series data. TCN excels in analyzing time-series data while GAN enhances training by generating realistic simulations. By combining the strength of both models, this approach aims to enhance stock price prediction accuracy. The proposed model uses TCN as the generator within the GAN framework and a Multilayer Perceptron (MLP) as the discriminator. TCN handles the prediction task and is trained using the GAN model. The model is trained over 500 epochs, with a learning rate of 0.0004 for the generator and 0.0001 for the discriminator. During each epoch, the generator is updated twice to enhance its performance. The resulting model achieves a MAPE score of 2.16% and an RMSE score of 814.25 on the testing dataset, demonstrating excellent performance in stock price prediction despite significant price variations.

Keywords: Generative Adversarial Network, Stock price prediction, Temporal Convolutional Network, Time-series data

INTRODUCTION

Stock market plays a crucial role in supporting national economies by enabling companies to secure funding and providing investors with opportunity to earn returns (Tambunan, 2020). However, these opportunities also come with risks due to the uncertainties in the market behavior. These uncertainties are often caused by factors such as inflation, time, and external economic conditions. Stock prices often go through big changes, showing just how unpredictable the market can be. For example, a stock's price might rise quickly during a certain period but then drop sharply soon after. These sudden changes make it difficult for businesses to plan and for investors to make decisions. This unpredictability shows how complex the stock market is and why both companies and investors face so many challenges when trying to succeed in it. (Nurwita, 2023).

Forecasting stock price is inherently difficult because of the complex and dynamic nature of the market. Traditional models often struggle with the non-linear dependencies in financial time-series data. In contrast, deep learning has proven to be a powerful solution, offering significant advancements in stock price prediction by effectively handling these complexities (Lara-Benítez et al., 2020). Recent research by William and Rarasati (2024) highlights the exceptional performance of deep learning models in stock price prediction. Similarly, Petra et al. (2024) present simulations showing how neural networks can provide insights into the economic conditions of specific companies.

Temporal Convolutional Network (TCN) is a type of deep learning framework derived from the classic Convolutional Neural Network (CNN), specifically created to handle time series data effectively. TCN could capture long-term dependencies while preserving the sequential nature of data. This feature offers an advantage over traditional methods like Recurrent Neural Networks (RNN) (Lara-Benítez et al., 2020). On the other hand, Generative Adversarial Network (GAN) provide a complementary approach by generating realistic synthetic data that closely resembles actual financial data. GAN involves two parts: a generator that creates fake data and a discriminator that evaluates its authenticity (Aggarwal et al., 2021).

A research done by Yujie et al. (2019) demonstrated outstanding results in various time-series predictions using the TCN model. In one experiment, TCN achieved a MAPE score of 2.96% in predicting Microsoft's stock

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 9, Number 1, January 2025 e-ISSN : 2541-2019
DOI : https://doi.org/10.33395/sinkron.v9i1.14246 p-ISSN : 2541-044X

price. They integrated a gated linear unit in the TCN model and successfully reduced the MAPE score by 0.02%. Both modified TCN and the original TCN performed exceptionally well across other time-series datasets, consistently achieving low RMSE, and MAPE values.

Another research developed by Kang et al. (2019) explored the use of GAN for stock price prediction. In the proposed GAN, the generator is an LSTM, and the discriminator is a simple MLP. The study result shows a strong performance on the GAN model, achieving a RMSE score of 4.1026, and MAPE score of 1.37%. The GAN model performance exceeds other deep learning algorithm such as LSTM, ANN and SVR.

Combining the strength of both TCN and GAN could offer a novel approach to stock price prediction. Previous studies have highlighted TCN's effectiveness in handling extended correlations in time-series data and GAN's ability to generate realistic synthetic data, yet their combined potential in financial forecasting remains underexplored. The proposed model leverages TCN's ability to handle sequential patterns and GAN's capacity to enrich datasets with realistic simulations. This study contributes to advancing forecasting methodologies and offers practical insights for investors navigating volatile stock market.

LITERATURE REVIEW

Deep learning itself is a part of machine learning that draws inspiration from the human brain structure, using multiple hidden layers in an artificial neural network to process data (Santoso & Ariyanto, 2019). One of the common methods used in deep learning is CNN. CNN is widely used for tasks involving spatial data, such as image recognition, and video analysis. By applying convolutional filters, CNNs effectively capture spatial hierarchies and patterns, making them a powerful tool for handling large and complex datasets in computer vision and other predictive tasks. CNN is commonly used for image processing, utilizing their specialized design to grid structured or 2-dimensional data (Sulaiman & Matius, 2023). A basic CNN typically includes layers such as dense layer, convolutional layer, dropout layer, pooling layer, and batch normalization layer.

TCN is a modification on CNN designed to work with 1-dimensional data. TCN uses a convolutional structure to handle tasks like time-series modeling or sequential data analysis without using loops or recurrence (Chan Hong et al., 2023). TCN is built on the idea of a 1-Dimensional Fully Connected Layer (1D-FCN), where each hidden layer has dimensions equal to the input layer. This is done by adding zeros, also known as zero-padding, to ensures the output length remains identical to the input length (Deng et al., 2019). TCN incorporates three key structures: Causal Convolution, Dilated Convolution, and Residual Block (Yujie et al., 2019).

A convolutional layer is designed to preserved the chronological order of data during the convolutional process. At time *t*, the output depends on the data before *t*, which prevents information leakage from the future.

$$y_t = \sum_{i=0}^{k-1} f_i \cdot x_{t-i} \quad (1)$$

Equation (1) represents the output of a convolutional layer (Bi et al., 2022). The output at time *t* is only affected by the data before time *t*, within the range defined by the kernel size *k*. The filter *f* determines the contribution of each input point in this range to the output. Each input value is multiplied by its corresponding filter weight, and the results are merged to obtain the output.

A large receptive field is important for understanding long-term patterns in time-series data. While stacking more convolutional layers or using bigger filters can achieve this, these methods require expensive computing cost. Dilated convolution solves this problem efficiently by using filters with gaps between input points. This expands the receptive field without adding more parameters or making the network deeper (Chan Hong et al., 2023).

$$y_t = \sum_{i=0}^{k-1} f_i \cdot x_{t-i \cdot d} \quad (2)$$

Equation (2) shows an additional variable *d* as the dilation rate on (1). Increasing the dilation rate *d* allows the network's receptive field to grow exponentially without needing larger filters or a deeper network (Yujie et al., 2019).

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 9, Number 1, January 2025 e-ISSN : 2541-2019
DOI : https://doi.org/10.33395/sinkron.v9i1.14246 p-ISSN : 2541-044X

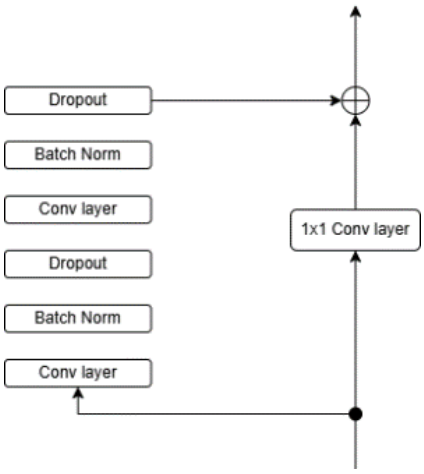

Fig. 1. Residual Block with a 1x1 Conv Input

A residual block in TCN helps prevent problems like vanishing or exploding gradients and stabilizes training using skip connections. (Chan Hong et al., 2023). A skip connection is a shortcut for the input to reach the output directly, without passing through a series of layers.

Fig 1 shows how skip connections work in a neural network. They allow the layers to focus on learning changes to the identity mapping, which is especially helpful in very deep networks (Bai et al., 2022).

$$o = Activation(x + f(x)) \quad (3)$$

The output of a residual block is defined in (3) (Chen et al., 2019). The output is determined by two inputs: the input processed through the layers within the block *f(x)*, and an input that passes through a 1x1 convolution as shown in Fig 1.

In a TCN, both input and output are 3-dimensional tensors. The input tensor is shaped as (batch_size, input_length, input_size), while the output tensor follows the shape (batch_size, input_length, output_size) (Sunitha et al., 2022). TCN ensures that the input and output lengths remain consistent across layers, with the only variation being in the third dimension, which corresponds to the input and output sizes.

GAN, found by Ian Goodfellow in 2014, consist of a generator and a discriminator, working against each other (Goodfellow et al., 2020). Unlike traditional methods, GAN doesn't rely on explicit probability distributions, making it flexible for capturing complex data patterns (Lan et al., 2020). The generator and discriminator work competitively. The generator continuously creating synthetic data while the discriminator verifies whether the data is original or synthetic. This adversarial process resembles a minimax game, where the generator tries to trick the discriminator, while the discriminator works to correctly detect the fake data (Goodfellow et al., 2020).

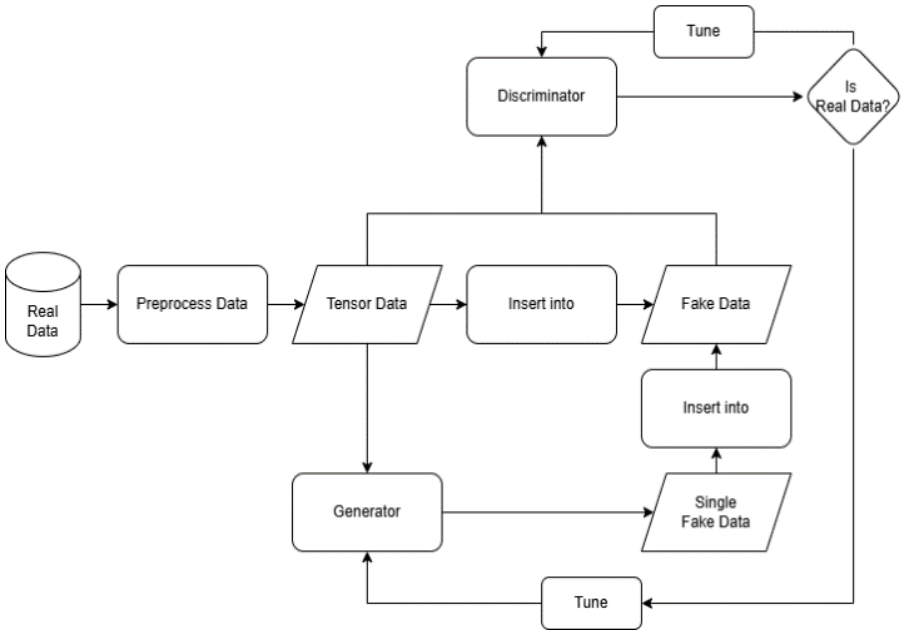

Fig. 2. GAN's Architecture

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 9, Number 1, January 2025 e-ISSN : 2541-2019
DOI : https://doi.org/10.33395/sinkron.v9i1.14246 p-ISSN : 2541-044X

In GANs, the main challenge is approximating the real data distribution, which is often complex and hard to compute directly. Traditional methods like Gaussian mixtures or maximum likelihood struggle with this complexity. The generator addresses this problem by generating synthetic data from a random variable sampled from a prior distribution. Its goal is to produce a distribution that closely matches the real data.

A discriminator is assigned to do classification task, which applies zeros for fake data and ones for real data. Binary Cross Entropy (BCE) is usually used to assist discriminator on its role. The loss function is defined in (4) (Lan et al., 2020).

$$Loss = -(y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})) \quad (4)$$

In (4), $\hat{y}$ represents the predicted probability of the data analyzed is real, and y is the actual label. By substituting the real and generated data distribution, the discriminator's objective will be defined in (6) (Jabbar et al., 2021).

$$V(G, D) = \mathbb{E}_{x \sim P_{data}(x)}[\log D(x)] + \mathbb{E}_{x \sim P_g}[\log(1 - D(G(x)))] \quad (5)$$

The first part of the equation (5) represents the expected value of the logarithm of the probability assigned by the discriminator to real data *x*, with *D(x)* being the probability that *x* is real. Conversely, the second part reflects the expectation for the generated data, where *D(G(x))* is the probability that the fake data appears to be real.

METHOD

The dataset used in this research consists of historical stock data for PT Indo Tambangraya Megah Tbk (ITMG), retrieved from Yahoo! Finance. The data covers the period from the company's initial listing on December 18, 2007, up to September 2, 2024. The dataset includes a total of 4,109 rows of data.

	count	mean	std	min	25%	50%	75%	max
Open	4109.0	2.473751e+04	1.152565e+04	4700.00000	15350.000000	2.440000e+04	3.285000e+04	5.680000e+04
High	4109.0	2.510374e+04	1.164145e+04	4815.00000	15550.000000	2.477500e+04	3.335000e+04	5.795000e+04
Low	4109.0	2.437016e+04	1.140235e+04	4675.00000	15075.000000	2.410000e+04	3.230000e+04	5.530000e+04
Close	4109.0	2.472202e+04	1.153274e+04	4730.00000	15300.000000	2.435000e+04	3.290000e+04	5.680000e+04
Adj Close	4109.0	9.421045e+03	6.862954e+03	1085.66687	4939.574219	7.696791e+03	1.006060e+04	3.248295e+04
Volume	4109.0	2.085164e+06	2.276700e+06	0.00000	876200.000000	1.478300e+06	2.513500e+06	4.545000e+07

Fig. 3. Statistic Descriptive of ITMG's Historical Data

Figure 3 presents the descriptive statistics of ITMG stock prices from 4,109 trading days. The average closing price is Rp 24,722.02 with a standard deviation of Rp 11,532.74, indicating a significant variation. The lowest price recorded is Rp 4,730.00, while the highest is Rp 56,800.00, with a median of Rp 24,350.00. About 25% of the prices are below Rp 15,300.00, and 75% are below Rp 32,900.00.

Before the training process, the dataset undergoes a preprocessing phase, includes splitting data, scaling data, and reshaping data. First, the data is divided into two sets with 80% for training and 20% for testing. In this study, MinMaxScaler is ideal for stock prices, which usually change within a predictable range. It normalizes data between 0 and 1 without changing temporal patterns or the original distribution (Altinbilek et al., 2022), Finally, the dataset is reshaped into the 3-dimensional format required by the TCN using the sliding window method. This method creates input-output pairs by using a fixed-size window to capture sequential dependencies in the data.

This research will use a TCN as the generator for the GAN model, and a simple Multilayer Perceptron Network (MLP) as the discriminator. The proposed model will be built in the structure shown in figure 4 and 5. The proposed TCN generator consists of three stacked dilated causal convolutional layers and one residual block. Each stacked layer features an exponentially increasing dilation rate (1, 2, 4 times) in the convolutional layer, accompanied by a batch normalization layer and a dropout layer. The residual block is used to keep the original input and add it back to the output of the stacked layers. The combined output from the stacked layers and the residual block is then passed through an average pooling layer, and a dense layer, to generate the final output. On the other hand, the MLP discriminator includes a sequence of dense layers, each activated by Leaky ReLU with a negative slope of 0.2. The output layer utilizes a sigmoid activation function, providing a probability between 0 and 1.

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 9, Number 1, January 2025
DOI : https://doi.org/10.33395/sinkron.v9i1.14246
e-ISSN : 2541-2019
p-ISSN : 2541-044X

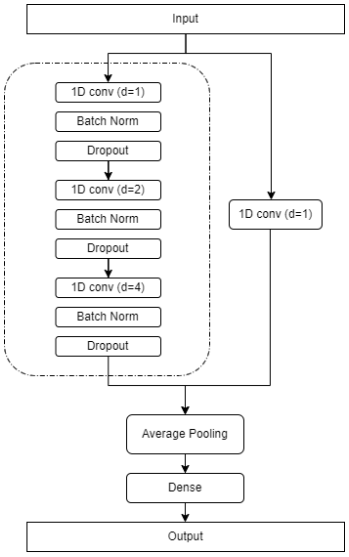

Fig. 4. Architecture of TCN

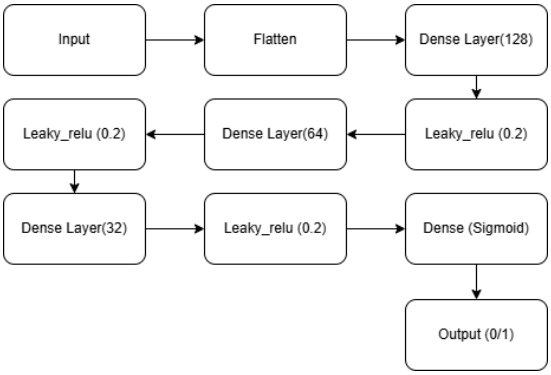

Fig. 5. Architecture of MLP

The GAN training process begins by defining the batch size and number of epochs, with 128 samples processed per batch and the training repeated for 500 epochs. Within each epoch, the generator creates synthetic data that tries to create data similar to the real thing, while the discriminator examines both real data and generated data to label them as real or fake. The training process within an epoch involves updates for the generator and discriminator. First, the discriminator calculates its loss by comparing its predictions for real and synthetic data against the true labels. It then updates its weights to improve its classification accuracy. Next, the generator computes its loss based on the discriminator's feedback, and then adjusts its weights to produce more realistic synthetic data that is harder for the discriminator to distinguish. This adversarial loop continues throughout the epochs.

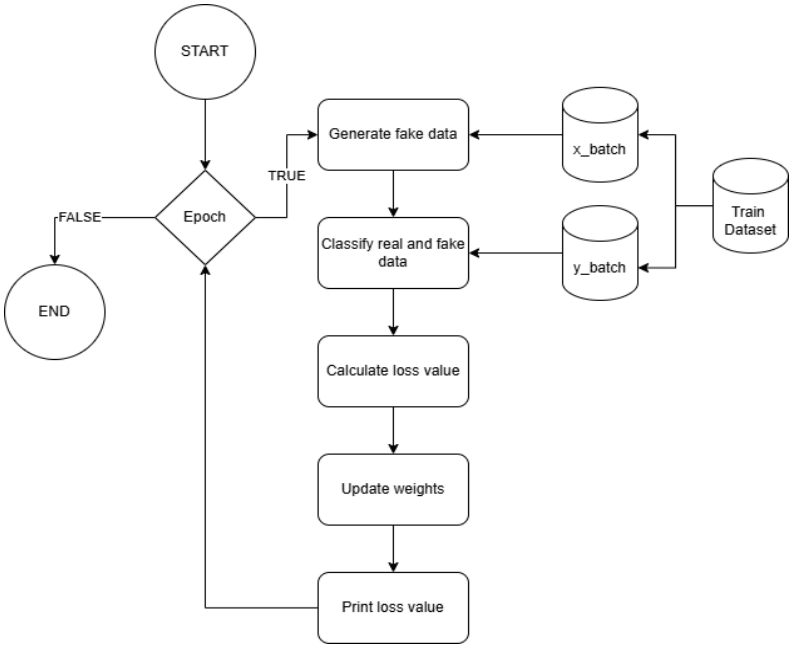

Fig. 6. Model's Training Process

After the training process, the results are evaluated using RMSE and MAPE. RMSE reflects the standard deviation between the predicted and actual values, showing a measure of how closely the predicted values align with the actual values (Hodson, 2022). RMSE can be defined as equation (6).

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 9, Number 1, January 2025
DOI : https://doi.org/10.33395/sinkron.v9i1.14246
e-ISSN : 2541-2019
p-ISSN : 2541-044X

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

The variable $\hat{y}$ represents the predicted value, y represents the actual value and n as the total row of data evaluated. RMSE measures the average size of prediction errors. A low RMSE shows better model performance. It is sensitive to large errors, making it useful for datasets with high variability.

As for MAPE, it shows prediction errors as a ratio of the actual values in percentage terms. MAPE can be defined as equation (7).

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100\% \quad (7)$$

The variable $\hat{y}$, y, and n represent the same thing as the RMSE equation (5). APE measures the prediction error as a percentage, making it useful when relative differences are more important than absolute errors A lower MAPE indicates higher prediction accuracy, especially for datasets where relative changes matter more than absolute values.

Budiprasetyo et al. (2023) categorize MAPE values into four levels: accurate, good, acceptable, and inaccurate, as shown in table 1. Models with MAPE below 20% are considered reliable with tolerable errors. MAPE between 20% and 50% indicates significant errors, suitable for rough analyses where precision is not critical. Models with MAPE above 50% are highly unreliable, as prediction errors exceed actual values, rendering the results unusable.

Table 1. MAPE Scale

MAPE	Category
< 10%	Accurate
10% - 20%	Good
20% - 50%	Acceptable
> 50%	Inaccurate

RESULT

Table 2. Hyperparameter Tuning

Epoch	Generator Learning Rate	Discriminator Learning Rate	Generator Updates per Epoch	RMSE	MAPE
200	0.0002	0.0001	1	4433,6	15,34
300	0.0002	0.0001	1	1633,67	4,45
400	0.0002	0.0001	1	1553,8	4,16
500	0.0002	0.0001	1	1181,7	3,27
500	0.0002	0.0001	2	905,69	2,32
500	0.0002	0.0001	4	1560,69	4,07
500	0.0002	0.0001	8	954,17	2,44
500	0.0003	0.0001	2	866,35	2,24
500	**0.0004**	**0.0001**	**2**	**814,25**	**2,16**
500	0.0001	0.0002	2	990,5	2,53

During the testing process, several parameters influence the modeling results, including the number of epochs, the learning rates of the generator and discriminator, and the number of updates applied to the generator in each epoch (Ngo, 2021). The combination of these four parameters will be tested in three groups. The first four test will modify the number of epochs, the next three tests will modify the number of updates to the generator in each epoch, and the last three tests will modify the learning rate ratio between the generator and the discriminator. The model will be evaluated using the testing dataset. The best result is achieved with a combination of 500 epochs, a learning rate of 0.0004 on the generator, 0.0001 on discriminator, and 2 generator updates per epoch.

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 9, Number 1, January 2025
DOI : https://doi.org/10.33395/sinkron.v9i1.14246
e-ISSN : 2541-2019
p-ISSN : 2541-044X

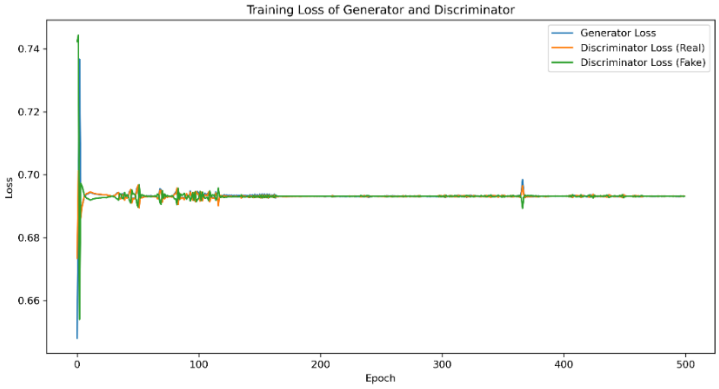

Fig. 7. Training Loss Graph

Fig. 6 shows the learning progress of both generator and discriminator throughout the training process. The training starts with a high difference of loss between the generator and discriminator. There are fluctuations on the beginning of the training. This shows the model is starting to learn. As the epoch increases, the fluctuations decrease. This indicates the model is beginning to capture the patterns while still continue to learn. By around 200 epochs, the losses stabilize, which means both models reached the equilibrium. There are still small fluctuations near the end of the training. This suggests the discovery of a new pattern. However, the model manages to stabilize the losses again. The stability on the final epochs is a positive sign that the model is functioning well.

Table 3. Evaluation on TCN-GAN

	Training Dataset	Testing Dataset
RMSE	857,81	814,25
MAPE	2,67%	2,16%

Fig. 8. Actual vs Predicted Data on Training Dataset

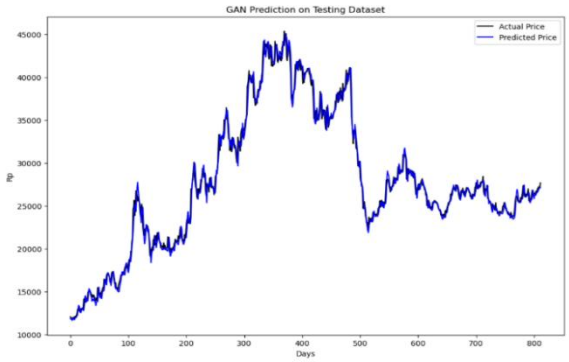

Fig. 9. Actual vs Predicted Data on Testing Dataset

Table 3 shows the evaluation of the trained model on both training dan testing dataset. In the training dataset, the model achieves a RMSE of 857,81 unit and a MAPE score of 2.67%. As in the testing dataset, the model performs better with a RMSE of 814,25 and a MAPE score of 2.16%.

Fig. 8 and fig. 9 visualize the comparison of the actual price and the predicted price on both training and testing dataset. Both figures show excellent alignments between the actual and predicted.

Table 4. Comparison on TCN dan TCN-GAN Results

	Training		**Testing**	
	TCN	**TCN-GAN**	**TCN**	**TCN-GAN**
RMSE	1507,95	857,81	1549,08	814,25
MAPE	4,92%	2,67%	4.12%	2,16%

The experiment is also conducted on the original TCN using the same structure for comparison. The original TCN is evaluated on both training and testing datasets. However, its performance does not surpass TCN-GAN. On the training dataset, the original TCN reaches a RMSE score of 1507.95 and a MAPE score of 4.92%. On the testing dataset, its performance yields a RMSE score of 1549.08 and a MAPE score of 4.12%.

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 9, Number 1, January 2025 e-ISSN : 2541-2019
DOI : https://doi.org/10.33395/sinkron.v9i1.14246 p-ISSN : 2541-044X

DISCUSSIONS

The results from table 3 indicates that the TCN-GAN model perform exceptionally well on both training and testing datasets. The difference in the results between both datasets is not significant, indicating the model generalizes effectively. This suggests that the model is neither overfitting nor underfitting.

Based on the MAPE score, the model is classified as highly accurate, because MAPE values below 10% are generally considered excellent for predictive performance. The RMSE score provides an indication of how closely the predicted values align with the actual values. In this case, the testing dataset achieves a RMSE of Rp. 814.25, which means the predicted stock prices deviate around Rp. 814.25. Considering the wide price range of stock dataset used in this research, which varies from Rp. 4730 to Rp. 56800 as shown in figure 3, the RMSE value indicates that the model has a relatively low prediction error. The better RMSE score on testing data also indicate that the model could perform accurately on an unseen dataset.

Table 5. GAN Comparisons on Previous Research

	RMSE	MAPE
LSTM (SBUX)	0,50522	195,319
Stacked LSTM-GAN (SBUX)	0,39562	213,377
LSTM (NKE)	0,55571	322,131
Stacked LSTM-GAN (NKE)	0,54009	139,925
TCN (ITMG)	1549,08	4,12
TCN-GAN (ITMG)	814,25	2,16

The experiment result also suggests the implementation of GAN successfully improve the performance of TCN. This improvement is proved by the significantly lower RMSE and MAPE scores. This shows greater accuracy and reliability in the stock price predictions. A similar research has been conducted by Ngo on a similar study case (Ngo, 2021). The results on the previous research is shown in table 5. Ngo utilizes the GAN model with LSTM as the generator. The previous research shows that a standalone LSTM has already performed exceptionally well on stock price predictions. By integrating GAN, it is reported that a significant improvement occurs in the predictive accuracy, because GAN could create a more realistic and robust prediction (Ngo, 2021). It is also said that GAN has successfully and effectively implemented on a stock price prediction study case. The current research is also reported in table 5, which also represent an improvement with the integration of GAN.

Although the results demonstrate a strong performance, the RMSE score reveals a serious concern. In stock price predictions, even small errors can be significant, as investors rarely purchase just one stock at a time. For instance, if the error margin is Rp 814.25 per stock, an investor buying 100 stocks could face a total discrepancy of Rp. 81425. This highlights the importance of minimizing prediction errors, as even seemingly minor inaccuracies can compound into substantial financial impacts when applied at scale.

This research relies only on historical stock price data, yet there are many factors that can influence stock market prices. Although GAN can provide a more realistic prediction, its accuracy may still be limited if these external variables are not incorporated into the model. Future studies could focus on incorporating these factors to enhance predictive accuracy and better understand the complex dynamics of the stock market.

CONCLUSION

The TCN-GAN model demonstrates strong performance in predicting the stock prices, achieving an RMSE of 814.25 on the training dataset and 857.81 on the testing dataset. These RMSE values reflect a relatively low average error between predicted and actual prices. Additionally, the model achieves excellent results in MAPE, with 2.67% on the training dataset and 2.16% on the testing dataset. This classifies TCN-GAN as a highly accurate predictive model. The implementation of GAN successfully improves the performance of TCN on stock price prediction.

Given the dynamic nature of the stock market, there are many external factors that can influence stock price trends. Future research could benefit from integrating relevant external variables, such as commodity prices, exchange rates, and interest rates, to help the model better understand the context that affects stock prices. Additionally, testing the model with data from periods of crisis or high volatility, such as the COVID-19 pandemic or the global financial crisis, could help the model better in adapting and maintaining accuracy during extreme market changes. Lastly, experimenting with variations of the TCN-GAN model, such as increasing the number of layers, epochs, or making other adjustments during the training process, could further improve its performance.

REFERENCES

Aggarwal, A., Mittal, M., & Battineni, G. (2021). Generative adversarial network: An overview of theory and applications. In *International Journal of Information Management Data Insights* (Vol. 1, Issue 1). Elsevier Ltd. https://doi.org/10.1016/j.jjimei.2020.100004

*name of corresponding author

Altinbilek, H. F., Nar, H., Aksu, S., & Kizil, Ü. (2022). Sensory Precipitation Forecast Using Artificial Neural Networks and Decision Trees. *Journal of Advanced Research in Natural and Applied Sciences*, *8*(2), 309–321. https://doi.org/10.28979/jarnas.984312

Bai, S., Zico Kolter, J., & Koltun, V. (2022). *An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling*. http://github.com/locuslab/TCN.

Bi, J., Zhang, X., Yuan, H., Zhang, J., & Zhou, M. C. (2022). A Hybrid Prediction Method for Realistic Network Traffic With Temporal Convolutional Network and LSTM. *IEEE Transactions on Automation Science and Engineering*, *19*(3), 1869–1879. https://doi.org/10.1109/TASE.2021.3077537

Budiprasetyo, G., Hani'ah, M., & Aflah, D. Z. (2023). Prediksi Harga Saham Syariah Menggunakan Algoritma Long Short-Term Memory (LSTM). *Jurnal Nasional Teknologi Dan Sistem Informasi*, *8*(3), 164–172. https://doi.org/10.25077/teknosi.v8i3.2022.164-172

Chan Hong, G., Nur Syazreen, A., & Patrick, G. (2023). TCN for transient simulation of highspeed channels. *2023 THE AUTHORS*, *1110–0168*. https://doi.org/10.1016/j.aej.2023.05.059

Chen, Y., Kang, Y., Chen, Y., & Wang, Z. (2019). *Probabilistic Forecasting with Temporal Convolutional Neural Network*. http://arxiv.org/abs/1906.04397

Deng, S., Chen, J., Zhang, N., Pan, J. Z., Zhang, W., & Chen, H. (2019). Knowledge-driven stock trend prediction and explanation via temporal convolutional network. *The Web Conference 2019 - Companion of the World Wide Web Conference, WWW 2019*, 678–685. https://doi.org/10.1145/3308560.3317701

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2020). Generative adversarial networks. *Communications of the ACM*, *63*(11), 139–144. https://doi.org/10.1145/3422622

Hodson, T. O. (2022). Root-mean-square error (RMSE) or mean absolute error (MAE): when to use them or not. In *Geoscientific Model Development* (Vol. 15, Issue 14, pp. 5481–5487). Copernicus GmbH. https://doi.org/10.5194/gmd-15-5481-2022

Jabbar, A., Li, X., & Omar, B. (2021). *A Survey on Generative Adversarial Networks: Variants, Applications, and Training*.

Kang, Z., Guoqiang, Z., Junyu, D., Shenke, W., & Yong, W. (2019). Stock Market Prediction Based on Generative Adversarial Network. *2018 International Conference on Identification, Information and Knowledge in the Internet of Things*, 400–406. https://doi.org/10.1016/j.procs.2019.01.256

Lan, L., You, L., Zhang, Z., Fan, Z., Zhao, W., Zeng, N., Chen, Y., & Zhou, X. (2020). Generative Adversarial Networks and Its Applications in Biomedical Informatics. In *Frontiers in Public Health* (Vol. 8). Frontiers Media S.A. https://doi.org/10.3389/fpubh.2020.00164

Lara-Benítez, P., Carranza-García, M., Luna-Romera, J. M., & Riquelme, J. C. (2020). Temporal convolutional networks applied to energy-related time series forecasting. *Applied Sciences (Switzerland)*, *10*(7). https://doi.org/10.3390/app10072322

Ngo, D. (2021). *Predicting Stocks with LSTM-based DRNN and GAN* [San Jose State University]. https://doi.org/10.31979/etd.fjsz-y926

Nurwita, N. (2023). Pengaruh Likuiditas dan Profitabilitas Terhadap Harga Saham Pada PT Indo Tambang Raya Megah Tbk. *JEMSI (Jurnal Ekonomi, Manajemen, Dan Akuntansi)*, *9*(2), 492–500. https://doi.org/10.35870/jemsi.v9i2.1073

Petra, S. M., Suryantara, I. G. N., & Tampinongkol, F. F. (2024). Prediksi Kebangkrutan Menggunakan Jaringan Saraf Buatan. *Jurnal Algoritma, Logika Dan Komputasi*, *7*(1), 661–676. https://doi.org/http://dx.doi.org/10.30813/j-alu.v2i2.6037

Santoso, A., & Ariyanto, G. (2019). Implementasi Deep Learning Berbasis KERAS untuk Pengenalan Wajah. *Jurnal Teknik Elektro*, *18*(01). https://www.mathworks.com/discovery/convol

Sulaiman, D. C., & Matius, T. M. S. (2023). Web-based Writing Learning Application of Basic Hanacaraka Using Convolutional Neural Network Method. *Ultimatics : Jurnal Teknik Informatika*, *15*(1).

Sunitha, G., Arunachalam, R., Abd-Elnaby, M., Eid, M. M. A., & Rashed, A. N. Z. (2022). A comparative analysis of deep neural network architectures for the dynamic diagnosis of COVID-19 based on acoustic cough features. *International Journal of Imaging Systems and Technology*, *32*(5), 1433–1446. https://doi.org/10.1002/ima.22749

Tambunan, D. (2020). Investasi Saham di Masa Pandemi COVID-19. *Jurnal Sekretari Dan Manajemen*, *4*(2). http://ejournal.bsi.ac.id/ejurnal/index.php/widyacipta

William, K., & Rarasati, D. B. (2024). Stock Price Prediction On IDX30 Index Using Long Short-Term Memory Algorithm. *JURNAL INFORMATIKA*, *11*(2), 80–89. https://doi.org/10.31294/inf.v11i2.22156

Yujie, L., Hongbin, D., Xingmei, W., & Shuang, H. (2019). Time Series Prediction Based on Temporal Convolutional Network. *2019 IEEE ICIS 2019*, *978-1-7281-0801-8/19/$31.00*.

*name of corresponding author