ISSN: 2993-8643

Engineering: Open Access

Review Article

Predicting Stock Prices Based on Price/Volume with Deep Learning and System Engineering Aggregate with Dynamic Behaviors and Trading Signals

Johannes K. Chiang[1] **and Renhe Chi***

[1]*Department of Management Information Systems, National Chengchi University.*

[*]**Corresponding Author**
Renhe Chi, Department of Management Information Systems, National Chengchi University Taiwan.

Submitted: 2023, Nov 01; **Accepted:** 2023, Dec 09; **Published:** 2023, Dec 10

Citation: Chiang, J. K., Chi, R. (2023). Predicting Stock Prices Based on Price/Volume with Deep Learning and System Engineering Aggregate with Dynamic Behaviors and Trading Signals. Eng OA, 1(4), 221-235.

Abstract
Current stock market forecasting methods encompass fundamental, technical, emotional, and bargaining factors. Predominantly, price prediction hinges on order volume and price, although correlating these two within existing models proves challenging. This study employs Cycle Generative Adversarial Network (Cycle GAN) to unravel the intricate price-volume relationship, combining it with Bollinger Bands for trading signal analysis, overcoming hurdles in short-term forecasting prevalent in numerical analysis and AI.

Focusing on TSMC (2330.TW) stock price, the research leverages Cycle GAN in deep learning to master the price-volume nexus, juxtaposed with LSTM and RNN. Historical TSMC closing prices and transaction counts are model inputs, scrutinizing their interconnectedness for predictions. This innovative approach aligns stock price, volume, market value, taxes, and prior changes via system engineering. By intertwining Bollinger Bands with stock price forecasts, trading signals are distilled, factoring in extended index %b for a comprehensive market picture.

In this research, under the framework of simulation system engineering, the stock price forecasted using RESNET yielded a 20.3% return, which represents a significant increase when compared to the original Bollinger Bands average return on investment of 15.5%.

Keywords: Deep Learning, System Engineering, Stock Price Forecasting, Aggregate Dynamic Behavior, Generative Adversarial Network

1. Introduction

TSMC, or Taiwan Semiconductor Manufacturing Company, is the largest global foundry with a commanding 58.5% share of the market. In advanced semiconductor processes, such as those with speeds of 5 nanoseconds (ns) or less, TSMC's market share surpasses 80%. Key clients of TSMC include industry giants like Apple, Qualcomm, Nvidia, and Intel, positioning the company as a prominent performer among tier 1 high-tech firms. Additionally, over 70% of TSMC's shareholders are international investors. The company is also listed as American depositary receipts (ADR) on the Nasdaq stock exchange [1].

The significance of TSMC in Taiwan's economic landscape is substantial. TSMC's equity represents nearly one-third of the total valuation of Taiwan's stock market [2]. Its annual capital expenditures make up 13% of private investment in the country, and its yearly production contributes approximately 5% to Taiwan's Gross Domestic Product (GDP). Moreover, TSMC plays a pivotal role in the semiconductor industry chain, stimulating growth among downstream manufacturers and facilitating the development of other Small and Medium-sized Enterprises (SMEs). TSMC's expansion is estimated to boost Taiwan's domestic output by NT$1.999 billion and create approximately 364,000 employment opportunities, indirectly influencing crucial economic metrics like the Consumer Price Index (CPI). Given these contributions, TSMC can be seen as the cornerstone of Taiwan's economic structure and a central focus for Taiwanese investors. Institutional investors predominantly control its equity distribution, while individual investors hold a relatively small fraction, accounting for less than 10%. Notably, the absence of a coherent and user-friendly investment metric presents challenges for retail investors, especially concerning capital and information disparities, which might lead to missed lucrative investment opportunities.

The advancement of deep learning techniques has witnessed rapid progression in recent years, offering analysis that often surpasses the objectivity of human intuition. Presently, the predominant architecture adopted for stock price prediction is the Long Short-

Term Memory (LSTM), which typically evaluates the volume and price of individual stocks in isolation [3][4]. Historical research underscores the intricate interplay between volume and price, suggesting a synergistic effect between them. Yet, conventional statistical models, relying primarily on the Correlation Coefficient, often misinterpret high covariance variables as independent, a misconception that has led to model inadequacies [5][6]. This study, utilizing TSMC stock as a case study, harnesses deep learning to scrutinize the interrelationship between stock prices and volume. By integrating these insights with stock price technical analysis, this research endeavors to offer a comprehensive suite of investment tools, bolstering investor confidence, especially in stocks with elevated unit prices.

This study aims to address the conjunction effect between stock prices and trading volume, with the objective of enhancing profitability rates:
1. Evaluation of the Cycle Generative Adversarial Network's efficacy in deciphering the aggregated relationships inherent in stock pricing, substantiating the model's learning proficiency [7].
2.A comparative analysis assessing the precision of Long Short-Term Memory and Residual Neural Network models in forecasting stock prices, utilizing insights gleaned from the aforementioned Cycle Generative Adversarial Network outcomes [8].
3. Formulation and assessment of a predictive model for short-term stock prices, employing an integrative approach that amalgamates system engineering principles with Bollinger Bands analytics [9].

2. Reference Review and Discussion

2.1. The Relationship Between volume and Price

The interrelation between trading volume and stock price delineates the nexus between a stock's transactional volume over a specified timeframe and its culminating price. Concurrent trajectories of volume and price suggest a definitive association between transaction volume and the amplitude of price fluctuations. Specifically, an augmentation in both stock price and transaction volume typically signifies prevailing market optimism. Conversely, a diminishing stock price accompanied by contracting volume suggests seller hesitancy and prospective future market buoyancy. Divergences between stock price and trading volume, where the two exhibit antithetical trends [10], bear distinct implications. For instance, a surging stock price with a static or diminishing trading volume may imply diminished investor endorsement, rendering the upward trajectory precarious. On the other hand, a declining stock price coupled with an escalating trading volume could be perceived as a harbinger of a bleak market forecast, prompting shareholders to liquidate. Whether the trajectories of volume and price are congruent or display deviations, their combined influence is palpable. Ying (1966) postulated that models solely predicated on stock prices and trading volumes are predisposed to yield erroneous or fragmentary outcomes.

2.2. Cycle Generative Adversarial Network (Cycle GAN)

The Generative Adversarial Network (GAN), as introduced by Ian et al. (2014), employs a binary classification strategy, utilizing an adversarial framework comprising both a Generative and a Discriminator model [11]. The Generative model, given random input values, can metamorphose these into visual representations via a deconvolutional neural network. In contrast, the Discriminator functions as a classification tool, discerning between authentic images sourced from the training dataset and fabricated images emanating from the Generative model. GAN's quintessential objective is to minimize a specific loss function, thus making the differentiation between synthetically produced and authentic images challenging. Owing to its design, which optimizes towards this end goal, GAN proves to be especially proficient for image synthesis tasks.

The transformation between images represents a complex visual and graphical challenge. Typically, GAN requires paired images within a training set to comprehend the mapping dynamics between the input and target images. However, in certain endeavors, paired training data might be unavailable. Addressing this challenge, the Cycle Generative Adversarial Network was introduced. This approach facilitates image transformation from domain X to domain Y under two critical stipulations: (1) the domains X and Y lack paired data; and (2) it is posited that a latent relationship exists between the X and Y domains.

In optimization scenarios involving standalone adversarial objectives that are chal lenging to optimize, disruptions during training can arise—specifically, all images might be mapped to a single output image, leading to optimization failures. The Cycle Generative Adversarial Network is architecturally premised on the existence of two transformation functions: $G(X) \rightarrow Y$ and $F(Y) \rightarrow X$. These functions, G and F, are conceptualized as inverses of each other, striving to achieve bijective relations. Through concurrent training of these mappings, G and F, a cycle consistency loss is introduced to ensure that $F(G(x))$ approximates x and $G(F(y))$ approximates y. By integrating this loss with the adversarial losses associated with domains X and Y, the network facilitates transformations between unpaired images.

2.3. Residual Neural Network (Residual Neural Network, ResNet)

Deep neural networks grapple with two predominant challenges. Firstly, there's the issue of Gradient Vanishing—where excessively small gradients during backpropagation hinder the model's ability to converge to an optimal solution. Secondly, there's the problem of gradient degradation, where errors accumulate as the gradient is backpropagated, leading to an escalation in error values as the network deepens. The Residual Neural Network introduces the Residual Block as depicted in Figure 1. Besides retaining the architecture of a conventional neural network, it integrates a shortcut connection. Assuming the desired output is $H(x)$ and the output of this particular layer in the original network is $F(x)$, if $H(x) = F(x) + x$, it's deducible that the output from this residual module, $H(x)$, can enhance the overall optimization of the neural network.

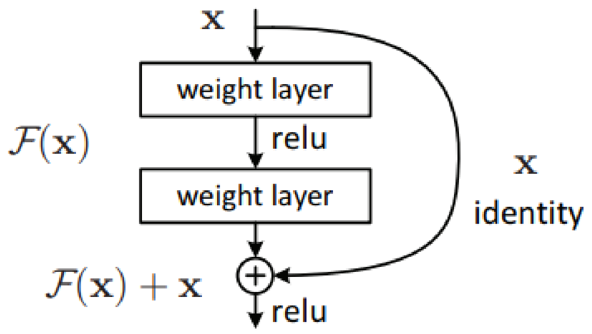

x
weight layer
relu
weight layer
$\mathcal{F}(\mathbf{x})$
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Figure 1: Residual Block

2.4. Long Short-Term Memory (LSTM)

During backpropagation training, Recurrent Neural Networks (RNNs) grapple with the challenges posed by evolving hidden states over time, notably the issues of gradient vanishing and gradient explosion. These obstacles impede their efficacy in handling long-term dependencies in time series data. Addressing this, the Long Short-Term Memory (LSTM) architecture, incorporates a gate mechanism into the RNN structure. This mechanism enables the model to retain input information over variable time lengths and modulates the model's input and output representations, effectively enhancing its capacity to process data with long-term dependencies.

2.5. System Engineering and Dynamic Behavior

Seely (1972) classifies system engineering models into three fundamental categories: Physical, Analytical, and Descriptive. Among these, the physical model stands out as the most conventional and prominent [12]. One merit of the physical model is its empirical verifiability, though it typically incurs higher costs. Descriptive models employ textual and graphical representations for decision-making purposes. In contrast, analytical models utilize mathematical techniques to depict system characteristics, often manifesting as sets of simultaneous equations. In this study, we employ various elements from the analytical model:

1. Mass: Invoking Newton's second law of motion, mass M is described by the equation: $f = M\frac{\mathrm{d}v}{\mathrm{d}t} = Ma$, v is the speed, *d* is the differential, *t* is the time, *a* is the acceleration.
2. Damper (friction force): Friction exhibits three primary mechanisms: static, Coulomb, and viscous. This research focuses on static friction, which pertains directly to motion. It emerges from the interaction at the point of contact between two surfaces: *f=Dv, Here,D stands for the friction constant,and v signifies velocity.*
3. Spring (Elastic Force): A spring is characterized as a component that conserves mechanical potential energy through its elastic deformation. *f=K∫vdt, In this context,K represents the spring constant,v is velocity,and ∫vdt is the integral of velocity concerning time.*

2.6. Bollinger Bands

Bollinger Bands (often abbreviated as BBands), sometimes referred to as Bollinger Channels or Bollinger Channels, are a technical analysis instrument introduced by John Bollinger. This methodology integrates the principles of moving averages and standard deviations. Fundamentally, it manifests as a channel delineated by three bands: a central band complemented by an upper and a lower band. The central band represents the average price of the stock, while the upper and lower bands can be interpreted as the stock price's resistance and support levels, respectively.

1. Bollinger band definition

[1] Middle band

The formula of the simple moving average for N time period is:

$$SMA = \frac{(P_1 + P_2 + P_3 + \cdots + P_n)}{n}$$

Where P is the stock price.

[2] Upper band

Standard deviation of the middle band + K × N time period

[3] Lower band

Middle track-standard deviation of K × N time period

2. Extended index──%b index

The position of the closing price in the Bollinger Bands is presented in digital form as a key indicator for trading decisions. The formula is:

%b= (close-lower band)/(upper band-lower band)

Band width= (upper band-lower band)/middle band

Given that the closing price fluctuates between the upper and lower bands, its amplitude can often exceed the band range (0~1), rendering the %b value unbounded. When there's an upward trend breach and the closing price surpasses the upper band, the %b value exceeds 1. Conversely, when the trend breaks downward and the closing price drops below the lower band, the %b value becomes less than 0.

Examining and analyzing the "%b indicator" can offer insights for investment considera-tions, enabling trading decisions predicated on the indicator's potency and nuances.

2.7. F1 Score Is A Way to Evaluate the Accuracy of The Test.

It takes into accounts the accuracy and recall in the test to calculate this score [13].

	The model predicts a rise	**The model predicts a fall**
The actual stock price is up	True Positive	False Negative
The actual stock price is down	False Positive	True Negative

Table 1: Confusion Matrix

Recall = TP / (TP+FN) represents the model's capacity to identify positive samples; a higher value indicates superior recognition capability.

Precision = TP / (TP+FP) signifies the fraction of accurately classified positive samples by the model. A value exceeding 0.5 indicates a robust capability to discern rising trends, while a value below 0.5 suggests a strong proficiency in identifying declining trends.
The F1 Score, defined as F1 = 2 * Precision * Recall / (Precision + Recall), serves as a harmonic mean of Precision and Recall. An F1 Score approaching 1 denotes optimal model classification performance.

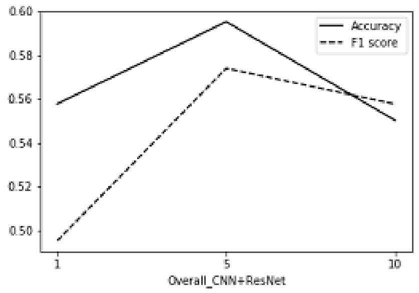


Figure 2: Overall CNN+RESNET training results

2.8. Imaging time series by GADF

In the realm of linear algebra, the Gram matrix—often referred to as the Gramian ma trix—pertains to a collection of vectors x 1,x 2,…,x N situated in an inner product space. This matrix is characterized as a Hermitian matrix, with its components being defined by the inner products of these vectors.:

$$Gij = (xi, xj)\ (i, j\ 2\ N),$$

$$G(x_1, x_2, \cdots, x_N) = \begin{bmatrix} (x_1, x_1) & (x_1, x_2) & \cdots & (x_1, x_N) \\ (x_2, x_1) & (x_2, x_2) & \cdots & (x_2, x_N) \\ \vdots & \vdots & \ddots & \vdots \\ (x_N, x_1) & (x_N, x_2) & \cdots & (x_N, x_N) \end{bmatrix} \quad , (1)$$

where (*xi, xj*) is the inner product of xi and *xj*. Given a time series data *X*, we rescale the time series data *X* so that all the values in *X* will fall in the interval [1, 1],

$$\tilde{x}_i^{[-1,1]} = \frac{(x_i - \max(X)) + (x_i - \min(X))}{\max(X) - \min(X)}. \quad . (2)$$

After the time series being rescaled, we get a rescaled time series data ˜*X*. Then we do the coordinate transformation. Change the time series data from Cartesian coordinate to polar coordinate. In the Cartesian coordinate, the time series data *Xi* is represented by time stamp and data value, that is (*ti*, ˜ *xi*). But in the polar coordinate, the time series data is represented by radius and angle, that is (*ri*, *ϕi*). We use the equations below to do the transformation, Upon rescaling the time series, a transformed time series data, denoted as ˜*X* , is procured. Subsequently, a coordinate transformation is executed, transitioning the data from the Cartesian coordinate system to the polar coordinate system. In the Cartesian framework, the time series data xi is delineated by a time stamp and its corresponding data value, represented as (ti, ˜ xi). Contrarily, within the polar coordinate system, the data is described by its radius and angle, given by (ri, ϕi) . The transformation is facilitated using the subsequent equations:

$$\begin{cases} r_i = \frac{t_i}{N} & t_i \in \mathbb{N} \\ \phi_i = \arccos(\tilde{x}_i) & \tilde{x}_i \in \tilde{X} \end{cases}, \quad (3)$$

where ti represents the time stamp, and N serves as a constant factor to normalize the scope of the polar coordinate system. In this study, N is set to 64, corresponding to the height and width of the time series image. The transformation represented by equation (3) exhibits two salient characteristics. Firstly, it is bijective, implying each time series uniquely maps to a singular transformation due to the monotonic nature of cosϕ when ϕ□[0,p]. Secondly, this transformation retains the absolute temporal relationships. Upon applying the coordinate transformation to the rescaled time series data, the resultant is the Gramian Angular Difference Field (GADF), described as follows [14]:

$$\mathrm{GADF} = [\sin(\phi_i - \phi_j)] = [\sin\phi_i \cos\phi_j - \cos\phi_i \sin\phi_j] = \sqrt{I - \tilde{X}^2}' \tilde{X} - \tilde{X}' \sqrt{I - \tilde{X}^2}, \quad (4)$$

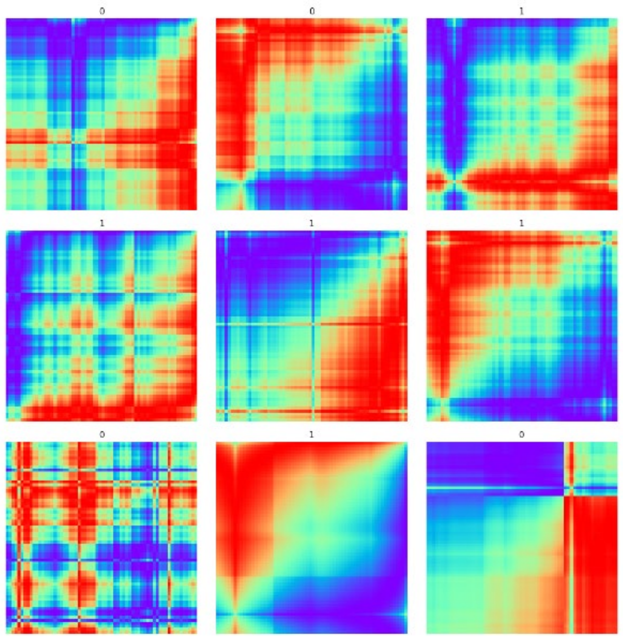
After getting the GADF, we define a new definition of the inner product as follows,

$$(x, y) = \sqrt{1 - x^2} \cdot y - x \cdot \sqrt{1 - y^2}.$$

At last, the GADF are can be regarded as the Gramian matrix because every entry in the Gramian angular fields is the inner product. The Gramian matrix is different from the traditional Gramian matrix in linear algebra for their definitions of inner product are different. Finally, with the different colors standing for different values of the entries in the Gramian angular fields, we can obtain the time series images.

Time Series Image

The generated time series images possess dimensions of 64×64×3 The objective is to employ the time series image, specifically the SSE Composite Index daily closing prices for the preceding 64 days, to forecast the direction of daily closing prices for the subsequent few days. An image receives a label of '1' if the average daily closing prices over the forthcoming 5 days is an upward trend compared to the previous 5 days; otherwise, it's labeled '0'. Ultimately, in the training and validation dataset, there are 3,314 images labeled '0' and 3,718 labeled '1'. In contrast, the test dataset contains 44 images labeled '0' and 51 labeled '1'.



GADF images with labels

3. Research Methods and Procedures

The research methodology was implemented using PyTorch, with the experimental procedures delineated as follows:

1. Development of a web crawler to gather historical data on TSMC stock, subsequently tored in a database. Data encompassing transaction volumes and closing prices from 2010 to 2020 were sourced from the Taiwan Stock Exchange (TWSE).
2. Data normalization.
3. Segmentation of data into training and test sets. Utilize the Cycle Generative Adversarial Network to discern the correlation between volume and price, subsequently constructing a model and calibrating training parameters.
4. Deployment of the cultivated model to forecast the test set data, integrating the price-volume relationship into both the residual neural network and the Long Short-Term Memory (LSTM) model. An assessment of the predictive accuracies of both models was undertaken.
5. Execution of short-term stock price forecasts, with an evaluation of forecasting outcomes achieved via dynamic simulation systems.
6. Synthesis of forecasting outcomes with technical analytical tools, specifically Bol linger Bands. The establishment of trading signals ensued, followed by an efficacy validation.

3.1 Data Set

1. Source: Taiwan Stock Exchange1
2. Stock Details: TSMC (2330.TW)
3. Data Span: 2010/01/04 - 2020/12/31, encompassing a decade of trading day records.
4. Historical Records: Closing price and daily trading volume.
5. Training Dataset: 90% of the aggregate data designated for training, with the remaining 10% allocated for validation.
6. Testing Dataset: The final 10% of the comprehensive data.

3.2 Data pre-processing

The trading volume's value spectrum differs from that of the stock's value range. Direct normalization using standard techniques might lead to significant disparities between these two values, potentially disrupting CycleGAN operations. To mitigate this, this research transforms the data into incremental changes, applies a logarithmic transformation to minimize discrepancies, and ultimately employs Min-Max normalization to confine the data range between 0 and 1. The Min-Max normalization formula is:

$$zi = \frac{xi - min(x)}{max(x) - min(x)}$$

zi represents the normalized value. Xi denotes the value undergoing normalization. min(x) signifies the data's minimum value, while max(x) corresponds to its maximum value.

To mitigate the risk of overfitting within the model, the dataset is partitioned into three segments. Specifically, 90% constitutes the training set, another 10% serves as the validation set, and the final 10% of the training data is allocated for testing purposes.

3.3 Deep Learning Neural Network Architecture

(1) Cycle Generative Adversarial Network Design

Prior research has highlighted a co-dependency between volume and price. Relying solely on one of these metrics for model computations will likely yield inaccurate outcomes. To grasp the interplay between volume and price, this investigation employs a CycleGAN for learning, central to which is the Cycle loss with the following loss function:

$$L_{cyc}(G,F) = E_{x\sim pdata(x)}\,[||F(G(x)) - x||_1] + E_{y\sim pdata(y)}\,[||G(F(y)) - y||_1].$$

For each image from the domain, execute the transformation reflecting the domain's combined influence of volume and price, specifically, revert to the original image via the cyclic mechanism, using the approach: X→G(X)→F(G(X))~~X.

Another core mechanism of the cyclic Generative Adversarial Networks is to combat loss, and its loss function is:

$$L_{GAN}(G, D_Y, X, Y) = E_{y\sim pdata(y)}[logD_Y(y)] + E_{x\sim pdata(x)}[log(1 - D_Y(G(x))]$$

Within this framework, G aims to render the generated image G(x) akin to an image from the Y domain, while Dy endeavors to discriminate between the transformed sample G(x) and the genuine sample from the Y domain.

The goals of this network are:

$$L(G, F, D_X, D_Y) = L_{GAN}(G, D_Y, X, Y) + L_{GAN}(F, D_X, Y, X) + \lambda L_{cyc}(G, F)$$

Finally to solve:

$$G^*, F^* = arg \min_{G,F} \max_{D_x,D_Y} L(G, F, D_x, D_Y)$$

In this study, price is considered the domain and transaction volume serves as the input for the cyclical generative adversarial network. FC stands for Fully Connective layer. Here, stock price and transaction volume undergo matrix reshaping, transforming them into two-dimensional matrix simulation data. This data is then utilized as the input for both the generative and discriminative models.

Given that CycleGAN was initially employed for mutual conversion in the image domain, the data is first transformed into two-dimensional simulate image data. It is then processed through the Convolutional Neural Network (CNN), renowned for its effectiveness in image tasks. This architecture, integrated with the residual neural network, serves as a generative model, producing outputs capable of deceiving the discriminative model. The discriminative model, utilizing a convolutional neural network framework, determines whether the two-dimensional data originates from training data or is fabricated by the generative model.

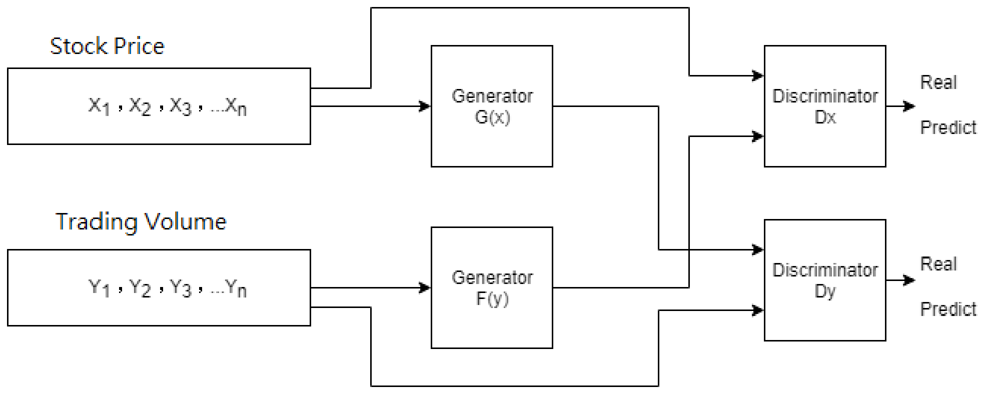


Figure 2: Cycle Generative Adversarial Network design diagram

Develop the residual neural network model: Feature extraction is facilitated through the convolutional neural network, which then links with the residual neural network. This employs 5 days of reconstructed stock prices and volume, with the residual neural network designed to encompass 48 layers.

Establish a Long Short Term Memory model (LSTM): Utilize the LSTM model to reconstruct time series data of volume and price. This model incorporates 5 days of restored stock price and volume, and its design encompasses 4 LSTM layers.

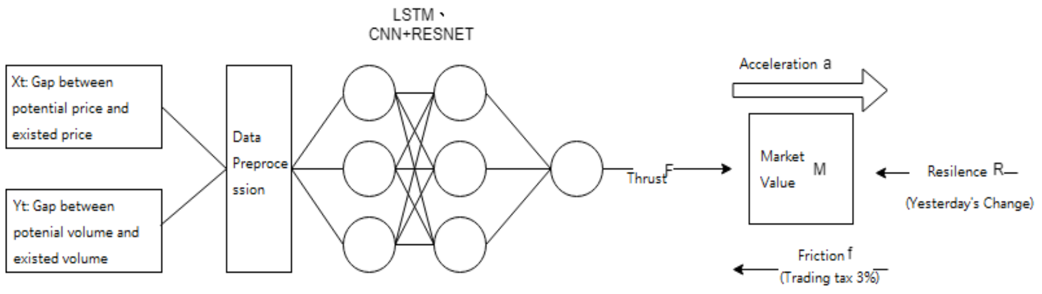


Figure 3: Prediction framework of stock price with simulation system engineering

(2) System engineering behavior and stock price prediction
Within the context of potential volume-price correlations leading to stock market variations, this study employs a system engineering paradigm, as illustrated in Figure 3. It suggests that the Cycle Generative Adversarial Network crafts approximated outputs G(x) and F(y) based on the anticipated future trajectory of volume and price. Shifts in the interplay between volume and price, perceived as the transformation from potential energy to kinetic energy, serve as the foundation for stock price alterations and dictate market dynamics. Specifically:

Propulsive Force (F): Xt, derived from simulated outputs F(y), is the difference between the potential and current price; Yt results from subtracting G(x) from the difference between potential and current quantities. Both Xt and Yt are fed into the neural network to replicate the force arising from the conversion of potential energy.

Market Capitalization (M): Market capitalization is computed as the product of stock price and shares outstanding. In the stock realm, price growth rate interrelates with market capitalization and capital influx. A uniform capital injection into varying market capitalizations results in distinct growth rates of stock prices. Market capitalization acts as a mass that's impelled by the propulsive force, simulating how uniform forces affecting disparate market values lead to varied stock price movements.

Drag (f): Represented as a transaction tax, drag is calculated as the product of transaction volume, stock price, and 0.3‰. In dynamic frameworks, drag correlates with the propulsive force. When viewed through the lens of stock market operations, it manifests as the transaction tax. Though the primary driver of stock price changes remains the volume of trade, the introduction of taxes in the market imposes a drag on transactions, justifying the portrayal of transaction tax as a form of drag.

Elasticity (R): Determined by the previous day's stock performance, a relevant value is assigned. The stock's elasticity pertains to its stored potential energy. Persistent effects of stock fluctuations greatly influence future prices. It's hypothesized that a steep surge in stock prices leads to a higher accumulation of potential energy, reducing the likelihood of further rises in the immediate future, and the opposite holds true for sharp declines.

Acceleration (a): Acceleration, denoted as 'a', is deduced from a distance equation, presuming an initial speed of zero, while 'k' stands for the stock price at the projected time point (in days). The governing equation is.

$$a=2(\text{data}_{t+k} - \text{data}_{t}) / t^2$$

3.4. Bollinger Band Design

Utilizing the extended %b indicator from the Bollinger Bands as a trading signal allows us to gauge the short-term strength of stock prices. This metric provides insight into whether the stock price currently stands at a relative recent high or low, thus facilitating informed trading decisions.

(a) Formulation of Bollinger Bands:
Middle Band: A simple moving average over an N-time period.
Upper Band: Middle Band plus K times the standard deviation over an N-time period.
Lower Band: Middle Band minus K times the standard deviation over an N-time period.
(b) Trading Strategy:
Take the day's closing price into account.
The parameter K for the three-band setting is determined experimentally using the Bollinger Bands. Both Experiment 1 and 2 aim to identify the optimal K value that yields the best average rate of return.
Similarly, the parameter N for the three-band setting is determined through experimentation with Bollinger Bands across experiments from 5 to 35, at 5-interval increments. The goal is to identify the N value that produces the most favorable average rate of return.
If the %b indicator exceeds or equals 1, the stock should be sold (with the stipulation that if it hasn't been previously purchased, it should not be sold).
Conversely, a stock should be bought when the %b indicator is less than or equal to 0.
(c) Integration with Stock Price Forecasting:
Incorporate the predicted stock price into the Bollinger Bands for the subsequent day, and discern the stock's current strength based on this future information. If the stock price is anticipated to increase, it can be inferred that its current position is not relatively high, allowing for potential delays in selling. The opposite strategy applies if the stock is predicted to decrease. The primary objective of this approach is to enhance investment returns.

4. Evaluation and Experimental Results

For predicting and assessing the stock price at the subsequent time point, we shall employ three distinct models:
Mean Squared Error (MSE): This metric computes the expected value of the squared deviations between the actual and predicted values. MSE serves a dual purpose: it assesses the model's accuracy and offers a more effective gradient for convergence in neural network-like structures. With a consistent learning rate, as the loss approaches 0, the gradient is anticipated to diminish.

Mean Absolute Error (MAE): This metric calculates the expected value of the absolute discrepancies between the real and forecasted values. The MAE offers a more accurate representation of the error's true magnitude between predicted and actual values.

Return On Investment (ROI): This metric signifies the percentage correlation between the investment returns and associated costs. Its formulation is as follows:

$$\text{ROI} = \left(\frac{\text{Investment Gain}}{\text{Investment Base}}\right)$$

4.1 CycleGAN learning results of volume-price relationship

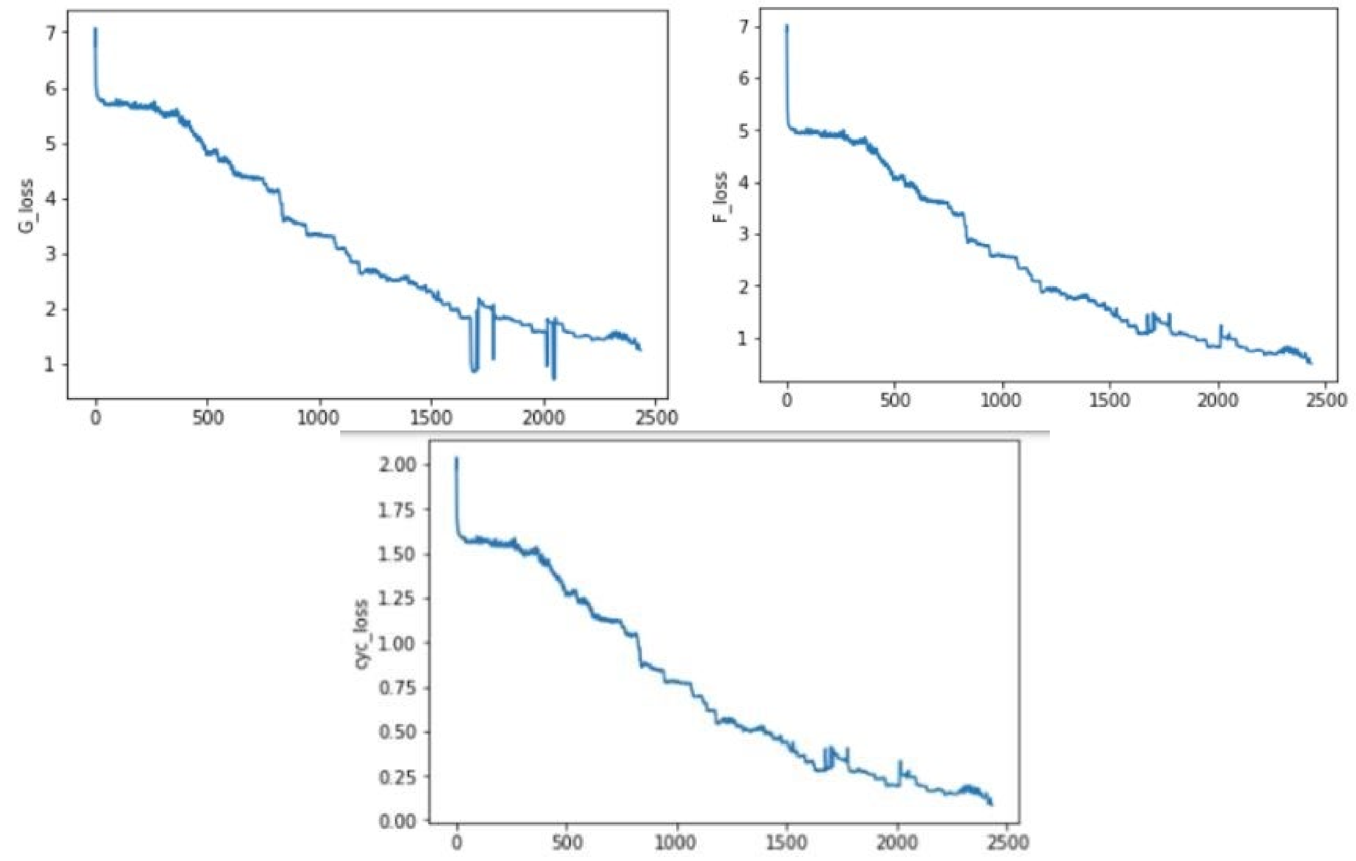


Figure 4: CycleGAN loss

From the observations, it is evident that as the number of epochs increases, the losses associated with Generator G, Generator F, and the cycle loss exhibit substantial reductions. It is hypothesized that the cycle loss functions as a hyperparameter and, upon reaching a certain threshold during training, ceases to undergo further training.

	Training Cycle loss	**Validation Cycle loss**	**Testing Cycle loss**
20 days	0.031995635	0.038789876	0.036866438
30 days	0.016229397	0.016477194	0.015846474
40 days	0.0015571207	0.031062838	0.030280465

Table 1: CycleGAN loss (MSE).

It is evident that the RESNET network architecture effectively captures the relationship between volume and price, as reflected by the substantial reduction in Cycle loss during testing. This suggests the capability of the cycle generative adversarial network to learn the interplay between volume and price. Utilizing a 30-day span for volume and price as input yields the minimal loss value. When the parameter n is set to 30, the cycle loss for training, validation, and testing is at its minimum, with only marginal variations.

	Training	Validation	Testing	Average
RESNET Without System engineering Without normalization target data	1.951394	3.345	5.097	3.464465
RESNET With simulation of system engineering Without normalized target data	3.211395	4.104	5.390600	4.235332
RESNET Without simulation of system engineering With Normalized target data	1.9069737	3.999800	5.346600	3.7511246
RESNET With simulation of system engineering With Normalized target data	15.592605	9.153	12.2104	12.318668

Table 2: Average gap (MAE) of RESNET's forecast stock price

LSTM Without simulation of system engineering Without normalized acceleration	1.995237	3.380000	5.080200	3.485146
LSTM With simulation of system engineering Without normalized target data	3.5186053	3.760200	5.308800	4.1958684
LSTM Without simulation of system engineering Without normalization target data	2.002026	3.356000	5.073000	3.477009
LSTM With simulation of system engineering With normalization target data	11.24382	13.5698	18.659800	14.49114

Table 3: LSTM predicts the MSE of stock prices

4.2 The Conclusion of The Stock Price Forecasting.

4.2.1 Normalization and Without Normalization.

In the context of linear regression neural networks, target data is typically normalized to prevent excessive loss, which could result in protracted convergence times or even non-convergence. However, upon normalizing the target data in this study, several issues arose:

a. Discrepancies in normalization restoration: Given that the target data is derived from day-to-day stock price differences (and in the simulated system engineering, the target data is divided by the square of the number of days), the values might become too minute. Consequently, post-normalization restoration may present discrepancies from the original data.

b. Minimal data discrepancy pre and post-normalization in linear regression: The original data range spans from -1.8 to 2.3, thus scenarios of elongated convergence times or non-convergence are unlikely.

4.2.2 The Performance of Stock Price Prediction

The efficacy of stock price prediction is contingent upon the choice of normalization versus non-normalization, coupled with the simulation of the dynamic engineering module.

a. Within the simulation system engineering framework, the RESNET deep neural network exhibits superior learning capabilities for target data when normalized, surpassing LSTM. However, in scenarios devoid of normalization or without target data normalization, LSTM outperforms.

b. Excluding the simulation system engineering, normalized target data decreases training loss but augments validation and test losses, resulting in subpar average outcomes. This impacts the results of RESNET more substantially. Conversely, LSTM remains largely unaffected, displaying minimal disparities in training, validation, and test losses.

c. On aggregate, when integrated with the simulation of system engineering, LSTM's performance surpasses that of RESNET. Its validation and test losses more closely mirror its training loss than its counterpart, boasting an average loss of 4.1958684, marginally better than RESNET's 4.235332. Yet, when the simulation system engineering component is omitted, RESNET's performance parallels LSTM's, with negligible variations in training, validation, and test losses. Regarding average loss, RESNET's figure of 3.464465 slightly edges out LSTM's 3.47009.

d. Result-wise, excluding the integration of simulation system engineering, the most optimal stock price prediction, with the smallest deviation, is achieved by RESNET combined with non-normalized target data, registering the best result of 3.464465.

4.3 Trading Signals and The Return Rate Forecast

4.3.1 Research problems of Bollinger Bands

With N set to 35 and K designated as 2, the optimal reporting rate is obtained as
follows:

Trading Signal	Training	Validation	Testing
Buy	93	38	25
Sell	185	41	58
Trading frequencies	93	38	18

Table 4: Number of Bollinger Bands Trading Signals

Training	Validation	Testing	Grand Mean
0.140362	0.207090	0.121204	0.155066

Table 5: Bollinger Bands Average of Return on Investment

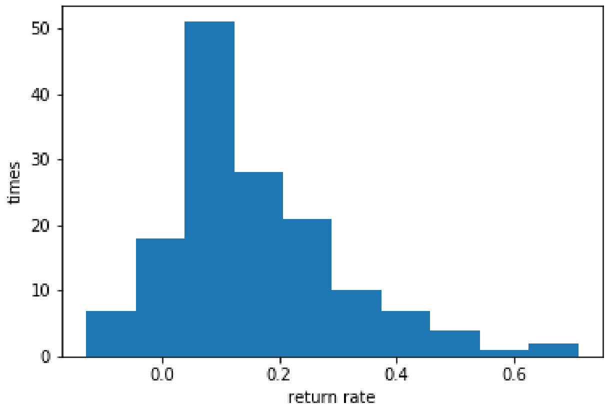


Figure 5: Bollinger Bands Investment Return Rate

4.3.2. Simulation of the integrated solution for stock prediction with LSTM and Bollinger band without System Engineering

Trading signals	Training	Validation	Testing
Buy	86	33	22
Sell	143	32	45
Trading frequencies	86	32	16

Table 6: Simulation of the integrated solution for trading signal with LSTM and Bollinger band without System Engineering

Training	Validation	Testing	Total average
0.191472	0.227625	0.123423	0.191980

Table 7: Simulation of the integrated solution for stock price prediction average ROI with LSTM and Bollinger band with System Engineering

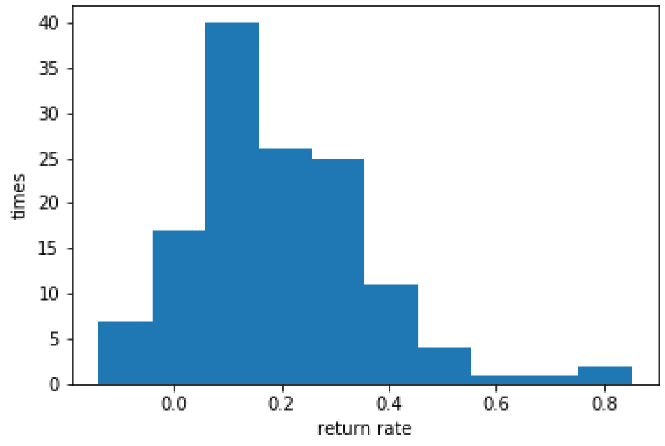


Figure 6: Simulation of the integrated solution for the ROI of stock price prediction with LSTM and Bollinger band with System Engineering

4.3.3. Simulation of the integrated solution for stock price prediction with RESNET and Bolling Band with System Engineering

Trading Signals	Training	Validation	Testing
Buy	84	33	22
Sell	132	32	47
Trading Signal	84	31	16

Table 8: Simulation of the integrated solution for price prediction and trading signal with ResNet and Bollinger Band with System Engineering

Training	Validation	Testing	Grand Mean
0.209931	0.228786	0.123423	0.203827

Table 9: Bollinger Bands and Simulation System Dynamics RESNET Predicted Stock Price Integration Average Return on Investment

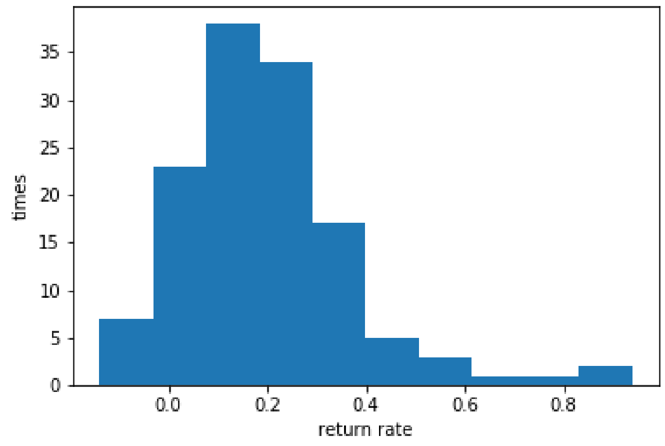


Figure 7: Simulation of the integrated solution for stock price ROI with RESNET and Bollinger Bang with System Engineering

4.3.4. Simulation of the integrated solution for stock price prediction with LSTM and Bollinger Bang without System Engineering

Trading Signal	Training	Validation	Testing
Buy	84	34	23
Sell	157	34	44
Trading Frequencies	84	32	17

Table 10: Simulation of the integrated solution for stock price prediction with trading signal with LSTM and Bollinger Band without System Engineering

Training	Validation	Testing	Grand Mean
0.156184	0.229696	0.124331	0.169800

Table 11: Simulation of the integrated solution for stock price prediction average ROI with LSTM and Bollinger Bang without System Engineering

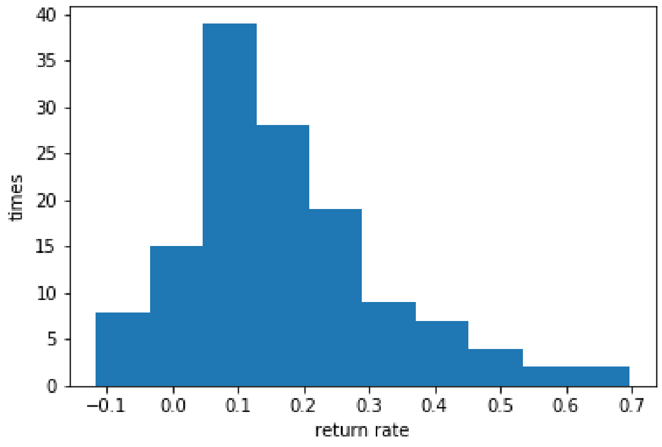
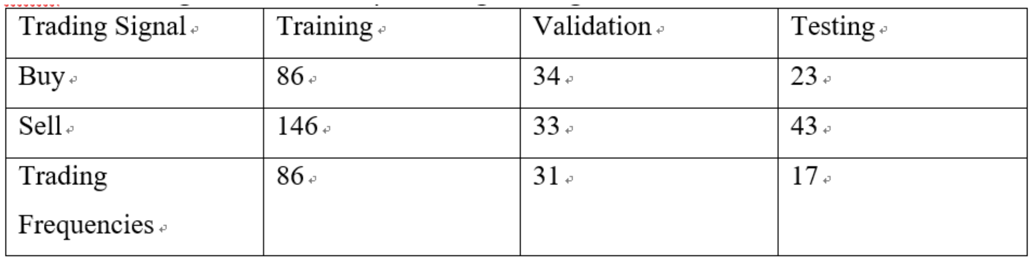


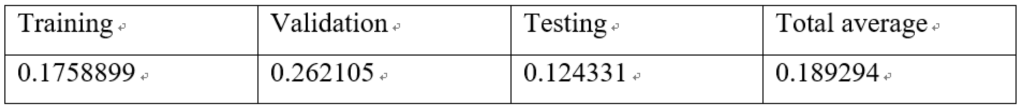
Figure 8: Simulation of the integrated solution for ROI with LSTM and Bollinger Bang without System Engineering

4.3.5. Simulation of the integrated solution for price prediction with RESNET and Bollinger Band without System Engineering



Trading Signal	Training	Validation	Testing
Buy	86	34	23
Sell	146	33	43
Trading Frequencies	86	31	17

Table 12: Simulation of the integrated solution for price prediction and trading signal with ResNet and Bollinger Band without System Engineering



Training	Validation	Testing	Total average
0.1758899	0.262105	0.124331	0.189294

Table 13: The average ROI for Simulation of the integrated solution for price prediction and trading signal with RESNET and Bollinger Band without System Engineering

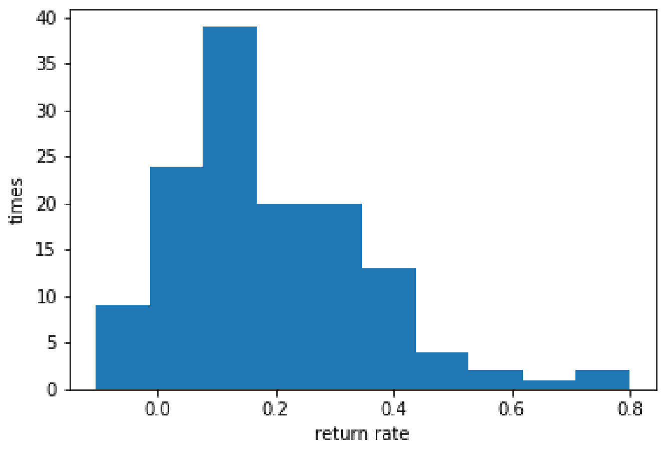
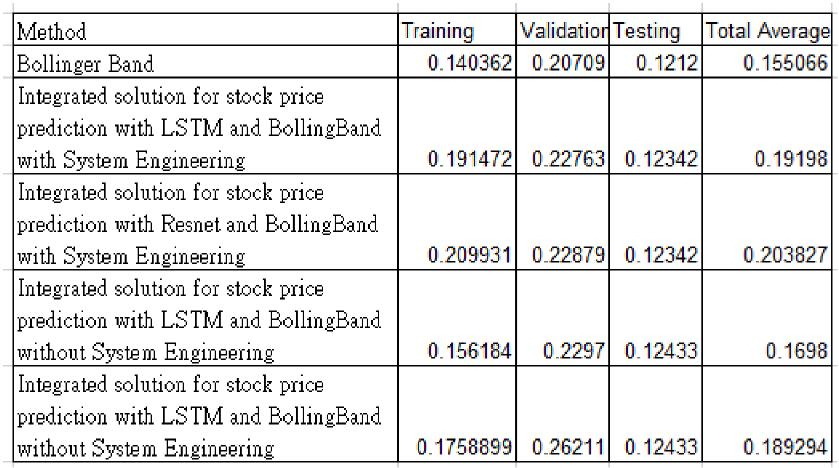


Figure 9: Simulation of the integrated solution for ROI with RESNET and Bollinger Bang without System Engineering



Method	Training	Validation	Testing	Total Average
Bollinger Band	0.140362	0.20709	0.1212	0.155066
Integrated solution for stock price prediction with LSTM and BollingBand with System Engineering	0.191472	0.22763	0.12342	0.19198
Integrated solution for stock price prediction with Resnet and BollingBand with System Engineering	0.209931	0.22879	0.12342	0.203827
Integrated solution for stock price prediction with LSTM and BollingBand without System Engineering	0.156184	0.2297	0.12433	0.1698
Integrated solution for stock price prediction with LSTM and BollingBand without System Engineering	0.1758899	0.26211	0.12433	0.189294

Table 14: Table of Return on Investment by Various Methods

5. Conclusion of Return on Investment

Bollinger Bands was an effective method originally, through its extended index %b, as the basis for buying and selling decisions, effective investment strategies can be derived. Notwithstanding the above, the return on investment can indeed be increased again through the integration with the forecast stock price. Under the simulation system engineering, the average return on investment after integration with the LSTM forecast stock price is 19.1%, and the average return on investment after integration with the RESNET forecast stock price is 20.3%; without the simulation system engineering, the average investment after integration with the LSTM forecast stock price, the return rate was 16.9%, and the average return on investment after integration with the RESNET forecast stock price was 18.9%. Both under the simulation system engineering and without system engineering integration with LSTM stock price forecast have return on investment higher than the original Bollinger Bands average return on investment of 15.5%. The propose of experimental design is accurate that the current stock price still represents a relatively high/low point at present and in the future through the integration of future information to make better decisions.

6. Limitation and Future Study

Addressing the limitations of this study, we propose the following directions for future enhancements and development:
Starting from the data perspective, collecting more international stock data can help in comparing the effects observed in both domestic and international markets. Additionally, exploring data from stocks with lower trading volumes domestically can further validate the generalizability of the proposed method.

In terms of trading strategies, this study primarily utilizes Bollinger Bands and its extended %b indicator as trading signals. Future research could explore integrating various trading signals with stock price predictions.

Regarding the neural network models and system dynamics simulations, while this study employs RESNET and LSTM for stock price predictions, the performance was outpaced by predictions solely based on neural networks. Future endeavors could consider integrating more parameters to enhance the accuracy of stock price predictions.

7. Research Conclusions and Recommendations

This research employs neural networks to forecast stock prices, drawing from TSMC stock data spanning from January 1, 2010, to December 31, 2020. It integrates volume-price relationships, system engineering principles, and Bollinger Bands to craft decision-making strategies, ultimately substantiating the efficacy of stock price predictions. The results highlight an enhanced return on investment when incorporating Bollinger Bands. Key findings include:

- This constitutes the inaugural study proposing the exploration of the latent volume-price relationship architecture. Through a Cycle Generative Adversarial Network that melds convolutional layers with residual networks, the research underscores the interconnected dynamics between volume and price. The innovative contribution of this research is reinforced by experimental validation of the aforementioned interconnectedness.
- Experimental evaluations with 20, 30, and 40 days of volume-price input into the Cycle Generative Adversarial Network reveal that a 30-day input garners the minimal cycle loss.
- By fusing the outcomes of Cycle Generative Adversarial Network learning with system engineering parameters, assumptions about potential volume-price deviations influencing future stock prices were posited and validated via system engineering simulations. Experimental analyses favored the latter approach, indicating superior stock price prediction accuracy.

References

1. Taiwan's TSMC controlled 60% of foundry market in Q1
2. Taiwan Economy Grows Fastest Since 2010 as TSMC Gives Boost
3. Gers, F. A., Schmidhuber, J., & Cummins, F. (2000). Learning to forget: Continual prediction with LSTM. Neural computation, 12(10), 2451-2471.
4. Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780.
5. C. C. Ying, Stock Exchange Listings and Securities Returns, Econometrica: Journal of the Econometric Society, 1966
6. Karpoff, J. M. (1987). The relation between price changes and trading volume: A survey. Journal of Financial and quantitative Analysis, 22(1), 109-126.
7. Zhu, J. Y., Park, T., Isola, P., & Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. In Proceedings of the IEEE international conference on computer vision (pp. 2223-2232).
8. He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).
9. Bollinger, J. (1992). Using bollinger bands. Stocks & Commodities, 10(2), 47-51.
10. Sheu, H. J., Wu, S., & Ku, K. P. (1998). Cross-sectional relationships between stock returns and market beta, trading volume, and sales-to-price in Taiwan. International Review of Financial Analysis, 7(1), 1-18.
11. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2014). Generative adversarial nets. Advances in neural information processing systems, 27.
12. Seely, S. (2013). An Introduction to Engineering Systems: Pergamon Unified Engineering Series (Vol. 9). Elsevier.
13. Townsend, J. T. (1971). Theoretical analysis of an alphabetic confusion matrix. Perception & Psychophysics, 9, 40-50.
14. Colla, V., Nastasi, G., & Matarese, N. (2010, November). GADF—Genetic Algorithms for distribution fitting. In 2010 10th International Conference on Intelligent Systems Design and Applications (pp. 6-11). IEEE.