VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY
UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING

——————— * ———————

PROJECT REPORT

Genetic Algorithm and Artificial Neural Network Hybrid Intelligence for Predicting Vietnam Stock Price Index Trend

COUNCIL: Computer Science
Supervisor: PhD. Nguyen Hua Phung
—o0o—
Student 1: Le Nhan Van (2252899)
Student 2: Nguyen Duc Tam (2252734)

HO CHI MINH CITY, 11/2024

Contents

Chapter 1

Introduction

1.1 Background

Predicting stock market movements, trends, and indexes is a complex challenge for researchers, as numerous factors influence these fluctuations. These factors include a company's growth potential and profitability, the local economic, social, and political environment, as well as global economic conditions. Accurate predictions are essential for reducing investment risks and optimizing returns.

The stock analysis process can be categorized into two types: fundamental and technical. Fundamental analysis evaluates the intrinsic value of a stock by examining key factors such as a company's growth potential, profitability, industry trends, and overall economic conditions. Conversely, technical analysis relies on mathematical methods using past stock index data. The simplest form involves observing stock movement trends on graphs, while more advanced approaches incorporate complex statistical techniques and machine learning algorithms. Among these, Artificial Neural Networks (ANNs) have gained popularity for time series forecasting and are widely recognized for predicting stock indices, trends, and market movements [3, 10]. In 1990, Kimoto et al. [14] pioneered the application of a modular neural network algorithm to forecast stock index movements on the Tokyo Stock Exchange and identify optimal buying and selling points. Over time, Artificial Neural Networks (ANNs) were further developed and became widely utilized in stock analysis. For instance, Wu and Lu. [22] applied ANN to forecast the S&P 500 stock index and compared its performance to predictions made by a Box-Jenkins model, finding that ANN provided more accurate results. Similarly, Guresen et al. [6] employed four models—ANN Multilayer Perceptron (MLP), Dynamic Architecture for Artificial Neural Network (DAN2), GARCH-MLP, and GARCH-DAN2—to predict the NASDAQ index. Their study concluded that the MLP model delivered the highest accuracy.

This paper will utilize the VN30 stock market index. VN30 is a list of stock market companies that officially came into effect on 6 February 2012 and was developed by the Index Committee, a group of independent financial experts established under the decision of HOSE. This committee is responsible for creating, compiling, and monitoring the index daily. The Index Committee reviews all components every six months to make necessary adjustments. Before its launch, extensive research was conducted in 2011, including trial operations and detailed consultations with numerous economic experts, to design the VN30 index. The VN30 is highly anticipated to provide a more accurate representation of the stock market.

As the name implies, VN30 comprises thirty stocks listed on HOSE. The selection of thirty stocks, instead of a different number, was based on an analysis of the market capitalization of representative stock samples. These thirty stocks collectively account for around 80% of the total market capitalization, with little variation compared to including fifty stocks.

1.2 Overview

Predicting stock market prices is a challenging task due to the complex and dynamic nature of financial markets. Various machine learning and statistical approaches have been used to tackle this problem, with Genetic Algorithms (GAs) emerging as a powerful technique due to their ability to optimize complex, multi-dimensional spaces effectively.

In this project, we will go through data input modifications for the model to be trained on. Setting up the ANN model to predict and calculate the accuracy of the prediction. Next, we focus on applying Genetic Algorithms to enhance the accuracy in the prediction of the trend of the stock market by minimizing the input features. Inspired by natural selection, GAs use mechanisms such as selection, crossover, and mutation to iteratively evolve a population of candidate solutions. Then we compare which input features will yield the highest ANN accuracy and in turn show which features should the investors be concerned about when it comes to making profits in the stock market.

The main contribution of this project is to improve the prediction of ANN and running time by minimizing input features. Moreover, the project also gives investors some insight into the past and current trends of Vietnam's stock market. These result in a more refined and educated choice in stock investment.

1.3 Literature Review

This review examines various studies that have applied ANN to predict stock prices and indices in both established and emerging markets. Leung et al. [15] utilized different models based on multivariate classification methods to forecast stock index trends. Their findings indicated that classification models, such as linear discriminant analysis, logit, probit, and probabilistic neural network (PNN), outperformed level estimation models, including exponential smoothing, multivariate transfer function, vector autoregression with Kalman filter, and multilayered feedforward neural network, in predicting the direction of stock market movement and maximizing return on investment trading.

Chen et al. [5] utilized a Probabilistic Neural Network (PNN) to predict the directional movement of the Taiwan Stock Exchange and leveraged these predictions to devise trading strategies. Their findings indicated that the PNN produced more accurate predictions compared to the GMM-Kalman filter and the random walk model. Similarly, Altay and Satman [2] contrasted the performance of Artificial Neural Networks (ANN) and linear regression in forecasting movement directions in emerging markets. Their results demonstrated that ANN achieved higher prediction accuracies of 57.8%, 67.1%, and 78.3% for daily, weekly, and monthly data, respectively.

Kara et al. [10] investigated the application of ANN and Support Vector Machines (SVM) to predict the movement direction of the Istanbul Stock Exchange (ISE) based on stock index data spanning 1997 to 2007. They used 10 technical indicators as input variables, including simple moving average, weighted moving average, momentum, stochastic K%, stochastic D%, RSI, MACD, Williams' R%, A/D oscillator, and CCI. Their results showed that the ANN model achieved prediction accuracies of 99.27% for training data and 76.74% for test data, while the SVM model attained 100% accuracy on training data but only 71.52% on test data.

Chang et al. [4] proposed an evolving partially connected neural networks (EPCNNs) model to predict stock price movements in the Taiwan Stock Exchange (TSE). Unlike traditional ANN, the EPCNN architecture used random neuron connections, supported multiple hidden layers, and optimized weights using genetic algorithms (GA). Their results showed that the EPCNN outperformed BPN, TSK fuzzy systems, and multiple regression analysis in prediction accuracy.

For Vietnam's stock market, Phuoc et al. [18] applied the Long Short-Term Memory (LSTM) algorithm with technical indicators, including the simple moving average (SMA), moving average convergence divergence (MACD), and relative strength index (RSI). Their dataset consisted of secondary data from the VN Index and VN-30 stocks. The study revealed that their LSTM model achieved high accuracy, with 93% prediction accuracy for most stocks, demonstrating its effectiveness in forecasting stock price movements using machine learning techniques.

Lastly, Inthachot et al. [9] explored a hybrid approach combining ANN and Genetic Algorithms (GA) to predict the trend of Thailand's SET50 index using data from 2009 to 2014. They utilized technical indicators as input variables, represented across 4 different time spans: 3, 5, 10, and 15 days before the prediction day. The hybrid method showed superior performance over a standalone ANN, with an average prediction accuracy of 63.60%, achieving an improvement of 12.40% on average.

For readers seeking a comprehensive review of recent advancements in stock market forecasting, Atsalakis and Valavanis [3] provide an excellent overview of the field.

1.4 Requirements and Objectives

1.4.1 Requirements

The input data for the VN30 index must consist of daily records spanning from January 1, 2019, to November 10, 2024 (which is 2140 days). Eleven technical indicators are calculated, each represented by four input variables corresponding to different historical time spans—3-day, 5-day, 10-day, and 15-day periods leading up to the prediction date. These inputs are designed to create diverse subsets of data, which are subsequently refined by a Genetic Algorithm (GA) to select the most effective features. The optimized input set is then fed into an Artificial Neural Network (ANN) to forecast the VN30 index trend.

Furthermore, due to the nature of GA, the time it took for the hybrid model to train is very long (4-5 days with CPU usage). So we would recommend using GPU to reduce the training time.

1.4.2 Objectives

Feature selection is inherently a multi-objective problem with the main objective of minimizing the number of features to maximize the classification accuracy of the ANN.

Another objective is to show the investors which features they should have in mind when considering stock market trading.

1.5 Problem Statement

Based on the dataset described partially in the Requirements section, the number of entries is 1,458 days due to some of the entries having missing values, and the number of features is 44 (11 indicators with 4 different time spans: 11 x 4 = 44). So we need to find a subset that minimizes the number of features to maximize the ANN's accuracy. Then based on the subsets that we got on different runs to generalize which features are important.

1.6 Outline

The remainder of this paper is organized into the following chapters: Chapter 2 describes the methodology including the research dataset, the preprocessing of the data, and the prediction models; Chapter 3 shows the experimental results and discussion; and Section 4 is the conclusion.

Chapter 2

Methodology

This chapter will first introduce and explain each individual's data preparation process and encoding scheme within the genetic algorithm used to optimize the neural network's parameters. Then, we will cover how we set up an appropriate ANN for the experiment. Finally, we will show the steps we took when building the GA algorithm (for the hybrid model of ANN and GA), which method we chose for each step, and how we evaluated the fitness value.

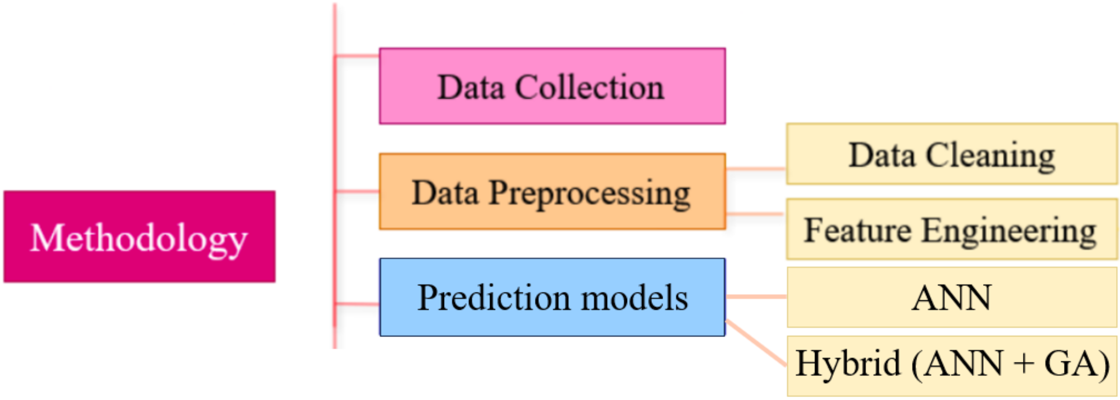


Figure 2.1: Methodology workflow for the project.

2.1 Data preparation and Preprocessing

This study utilized a dataset comprising daily closing values of a single company named ACB in the VN30 index recorded between January 1, 2019, and November 10, 2024, spanning 2,140 days. After calculating the 11 technical indicators (Figure 2.3) with 4 different periods and cleaning the dataset there are only 1409 days left. Within this timeframe, the stock index increased on 666 occasions (47.3%) and decreased on 743 occasions (52.7%), as detailed in Table 2.1.

The dataset was split into five groups for 5-fold cross-validation, as illustrated in Table 2.2. In each of the five runs, one group served as the test dataset, while the remaining four groups were used for training. This process ensured that every group was utilized as a test dataset exactly once.

From the widely recognized 11 technical indicators commonly used for forecasting stock prices and indices [8, 10, 12, 17], each input variable was computed using the corresponding

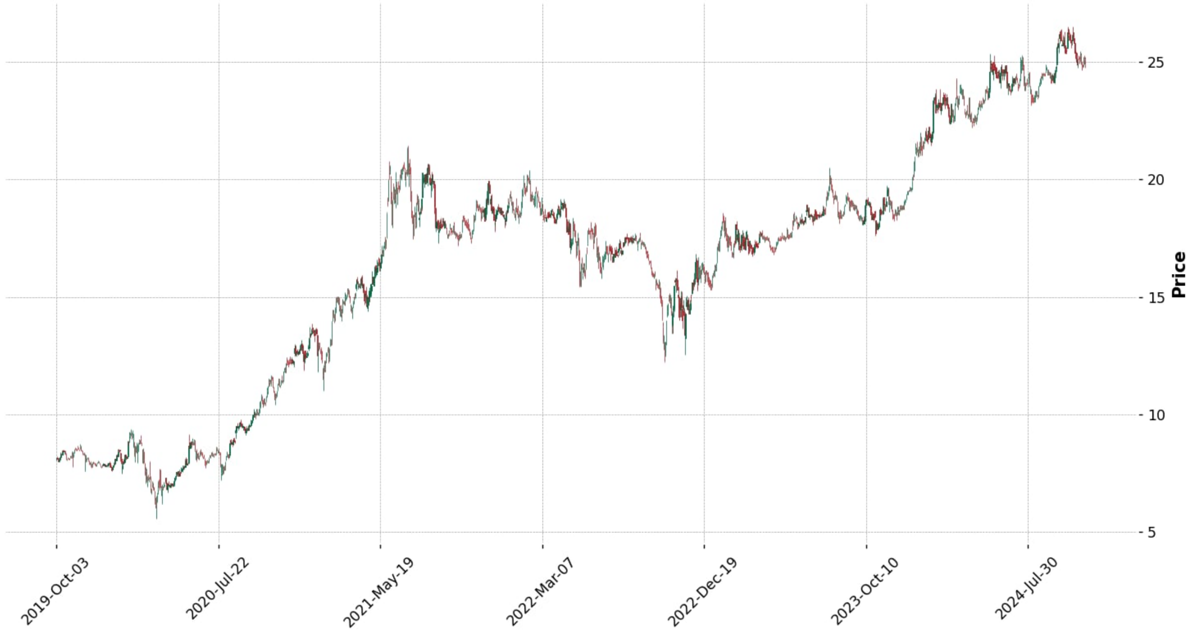
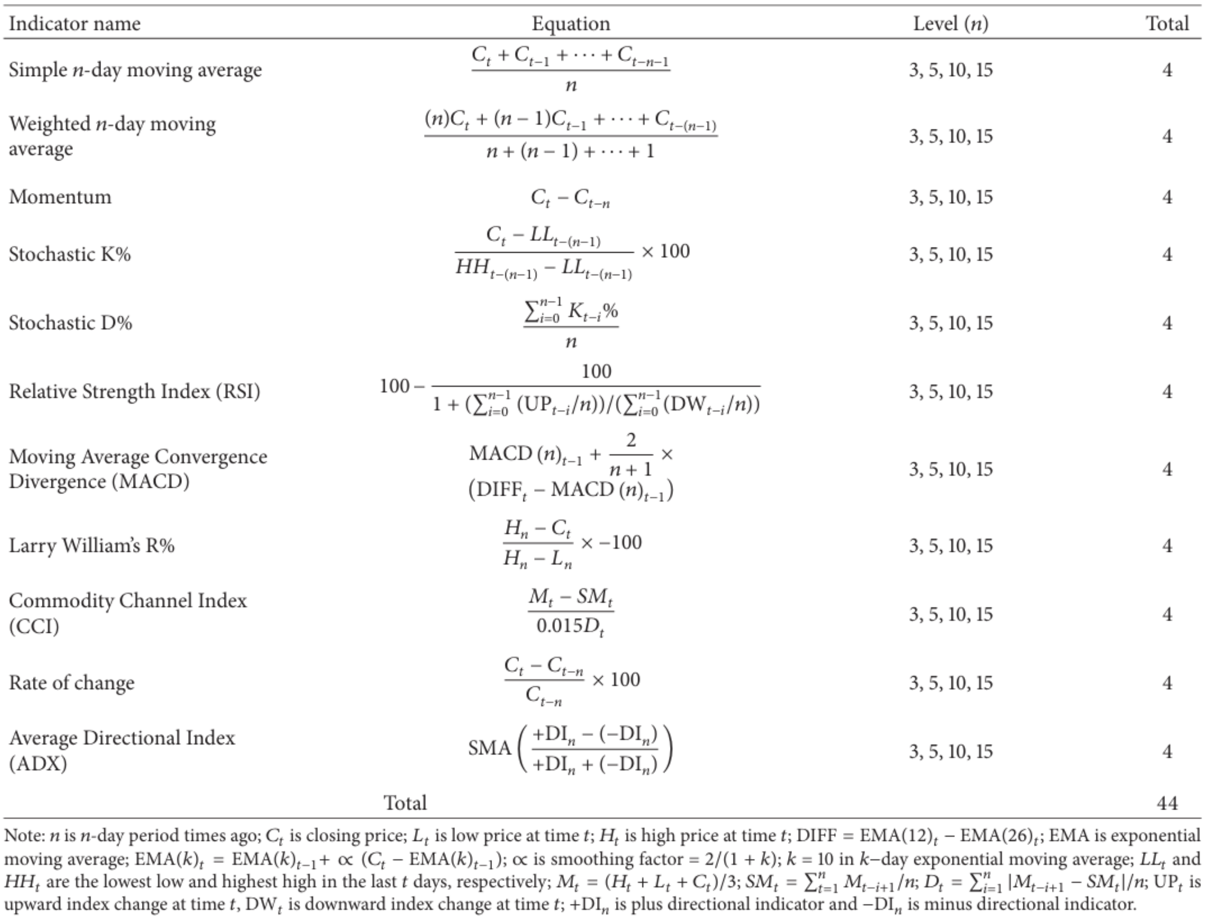


Figure 2.2: ACB stock market price from January 1, 2019, to November 10, 2024



Indicator name	Equation	Level (n)	Total
Simple n-day moving average	$\frac{C_t + C_{t-1} + \cdots + C_{t-n-1}}{n}$	3, 5, 10, 15	4
Weighted n-day moving average	$\frac{(n)C_t + (n-1)C_{t-1} + \cdots + C_{t-(n-1)}}{n + (n-1) + \cdots + 1}$	3, 5, 10, 15	4
Momentum	$C_t - C_{t-n}$	3, 5, 10, 15	4
Stochastic K%	$\frac{C_t - LL_{t-(n-1)}}{HH_{t-(n-1)} - LL_{t-(n-1)}} \times 100$	3, 5, 10, 15	4
Stochastic D%	$\frac{\sum_{i=0}^{n-1} K_{t-i}\%}{n}$	3, 5, 10, 15	4
Relative Strength Index (RSI)	$100 - \frac{100}{1 + (\sum_{i=0}^{n-1} (\mathrm{UP}_{t-i}/n))/(\sum_{i=0}^{n-1} (\mathrm{DW}_{t-i}/n))}$	3, 5, 10, 15	4
Moving Average Convergence Divergence (MACD)	$\mathrm{MACD}\,(n)_{t-1} + \frac{2}{n+1} \times (\mathrm{DIFF}_t - \mathrm{MACD}\,(n)_{t-1})$	3, 5, 10, 15	4
Larry William's R%	$\frac{H_n - C_t}{H_n - L_n} \times -100$	3, 5, 10, 15	4
Commodity Channel Index (CCI)	$\frac{M_t - SM_t}{0.015D_t}$	3, 5, 10, 15	4
Rate of change	$\frac{C_t - C_{t-n}}{C_{t-n}} \times 100$	3, 5, 10, 15	4
Average Directional Index (ADX)	$\mathrm{SMA}\left(\frac{+\mathrm{DI}_n - (-\mathrm{DI}_n)}{+\mathrm{DI}_n + (-\mathrm{DI}_n)}\right)$	3, 5, 10, 15	4
	Total		44

Note: n is n-day period times ago; C_t is closing price; L_t is low price at time t; H_t is high price at time t; DIFF = $\mathrm{EMA}(12)_t - \mathrm{EMA}(26)_t$; EMA is exponential moving average; $\mathrm{EMA}(k)_t = \mathrm{EMA}(k)_{t-1} + \propto (C_t - \mathrm{EMA}(k)_{t-1})$; $\propto$ is smoothing factor = $2/(1+k)$; $k = 10$ in k−day exponential moving average; LL_t and HH_t are the lowest low and highest high in the last t days, respectively; $M_t = (H_t + L_t + C_t)/3$; $SM_t = \sum_{t=1}^{n} M_{t-i+1}/n$; $D_t = \sum_{i=1}^{n} |M_{t-i+1} - SM_t|/n$; UP_t is upward index change at time t, DW_t is downward index change at time t; $+\mathrm{DI}_n$ is plus directional indicator and $-\mathrm{DI}_n$ is minus directional indicator.

Figure 2.3: Technical indicators used in this study and their equations [9, 10, 12]

Year	Up (times)	Up (%)	Down (times)	Down (%)	Total
2019	90	41.3%	128	58.7%	218
2020	120	49.4%	123	50.6%	243
2021	126	50.4%	124	49.6%	250
2022	119	48.0%	129	52.0%	248
2023	114	47.1%	128	52.9%	242
2024	97	46.6%	111	53.4%	208
Total	666	47.3%	743	52.7%	1409

Table 2.1: Yearly Up and Down distributions with percentages and totals

Year	Five runs of cross-validation										Total
	1st run		2nd run		3rd run		4th run		5th run		
	Up	Down	Up	Down	Up	Down	Up	Down	Up	Down	
2019	18	25	18	25	18	25	18	25	18	28	218
2020	24	24	24	24	24	24	24	24	24	27	243
2021	25	24	25	24	25	24	25	24	26	28	250
2022	23	25	23	25	23	25	23	25	27	29	248
2023	22	25	22	25	22	25	22	25	26	28	242
2024	19	22	19	22	19	22	19	22	21	23	208
Total	131	145	131	145	131	145	131	145	142	163	1409

Table 2.2: Yearly Up and Down distributions across five folds of cross-validation

indicator's formula, as outlined in Figure 2.3. Four input variables were derived from each technical indicator, with each variable calculated based on one of four historical time spans: 3, 5, 10, and 15 days. This resulted in a total of 44 input variables (11 × 4 = 44).

To ensure uniform weighting, all input variables were normalized to a range of [-1, 1]. The sole output variable was binary, taking a value of either 0 or 1. A value of 0 indicated that the predicted VN30 index for the next day was lower than the current day's index (downtrend), while a value of 1 signified that the predicted index for the next day was higher (uptrend).

2.2 Prediction models

2.2.1 Artificial Neural Network (ANN)

Artificial Neural Networks (ANN), first introduced by McCulloch and Pitts [16], are machine learning models designed to replicate aspects of human learning by utilizing past experiences to predict future outcomes. ANN has been extensively employed in research focused on forecasting stock prices and indices [3, 4, 6, 10]. Notably, it has also been applied specifically to predict trends in the VN30 index [8, 9]. Our ANN model was a three-layer feedforward architecture comprising an input layer, a hidden layer, and an output layer. Historical stock trading data were represented using 11 technical indicators, each of which was input into the ANN as 4 variables derived from 4 distinct historical periods. This configuration resulted in a total of 44 input variables in the input layer. The hidden layer contained 100 neurons, as determined to be optimal in the study by Inthachot et al. [8]. The transfer functions between the input and hidden layers, as well as between the hidden and output layers, were tan-sigmoid functions. The output layer consisted of a single neuron with a log-sigmoid transfer function. The output values ranged between 0 and 1, where values less than or equal to 0.5 indicated a downward index movement, and values greater than 0.5 indicated an upward movement. Weights were assigned to each connection between nodes, initialized randomly, and adjusted during training using the gradient descent with momentum method.

The parameters of the model requiring configuration included the number of hidden layer neurons (n), learning rate (lr), momentum constant (mc), and the number of training iterations (ep). These were set to $n = 100$, $lr = 0.1$, $mc = 0.1$, and $ep = 8000$, based on the settings that achieved optimal accuracy in the study by Inthachot et al. [8]. However, due to the time complexity, the number of iterations (ep) was reduced to 500 to accommodate training time constraints.

2.2.2 A Hybrid Intelligence of ANN and Genetic Algorithm (GA)

Artificial Neural Networks (ANN) have several limitations, including long training times, a tendency to converge to local rather than global optima, and a large number of parameters to configure. To address these drawbacks, researchers have explored hybridizing ANN with other algorithms that address specific issues. A commonly used algorithm in such hybrids is the Genetic Algorithm (GA). In 1990, Whitley et al. [21] utilized GA to optimize weight connections and design effective architectures for neural network connections. Later, Kim [13] proposed a hybrid ANN-GA model for instance selection to reduce data dimensionality. In 2012, Karimi and Yousefi [11] applied GA to determine optimal connection weights in an ANN model for analyzing nanofluid density correlations. Sangwan et al. [19] combined ANN and GA for predictive modeling and parameter optimization in turning operations to minimize surface roughness. Other successful applications of ANN-GA hybrids include network intrusion detection [20] and cancer patient classification [1]. Inspired by these successes, this study employs GA to address the feature selection problem, identifying effective subsets of inputs for ANN.

The rationale for adopting an ANN-GA hybrid approach in this study is twofold. First, multiple input variables (4 in this case) were derived for each technical indicator using different past time spans (3, 5, 10, and 15 days). Second, only a small number of effective subsets of input variables were selected for use. Considering the vast number of possible subsets of 44 variables (2^{44}), the computational effort required to evaluate all subsets would be impractical. GA is particularly effective for feature selection, and it was used here to identify optimal subsets of input variables.

Genetic Algorithm (GA) is a search algorithm inspired by natural selection and genetics, formally introduced by Holland in the 1990s [7]. The key principles of GA involve generating an initial population of chromosomes (candidate solutions) and iteratively applying selection and recombination operators to create new populations. Over successive iterations, the population evolves to include the fittest chromosome, representing the optimal solution.

There are 10 different steps when it comes to running the hybrid algorithm of ANN and GA. Figure 2.4 shows the overview diagram of these 10 steps.

Step 1: Population initialization

The initial population in this study was represented as a matrix with dimensions of Population Size × Chromosome Length, consisting solely of randomly generated binary digits. Here, Population Size refers to the number of chromosomes (or individuals) in the population, while Chromosome Length (or Genome Length) denotes the number of bits (or genes) in each chromosome. To ensure enough coverage of the search space, it is generally advisable to set the

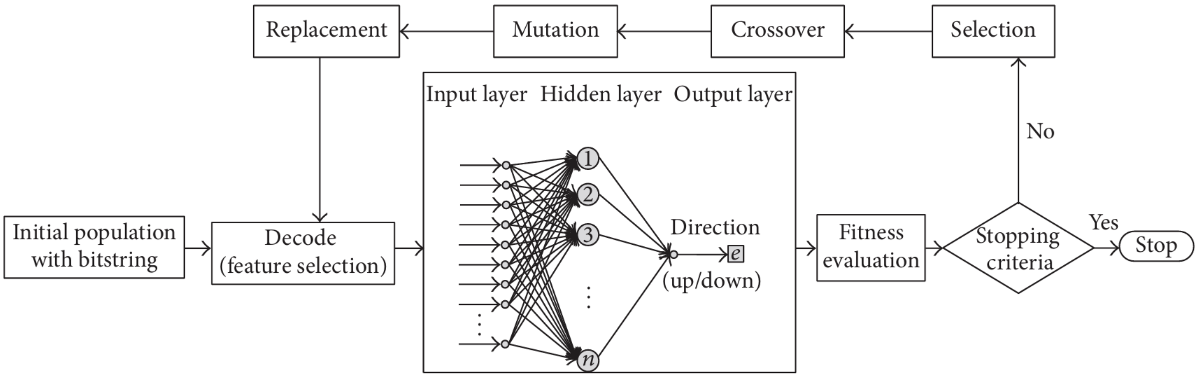


Figure 2.4: Steps of running the ANN + GA hybrid algorithm.

population size to be at least equal to the chromosome length. In this work, the Chromosome Length was set to 44, and the Population Size was chosen as 50.

Step 2: Decode (Feature selection)

Decode the chromosomes (bit strings) to determine which input variables are inserted into the ANN. For example, if the selected features are [2 3 5 6 8 10 11 14 15 16 18 19 21 22 23 24 27 36 37 38 39 40] then these 12 columns will be the input variables for the ANN.

Step 3: Artificial Neural Network (ANN)

Run a three-layered feedforward ANN model to predict the next-day VN30 index of the ACB company. The parameters in the model that we used were mostly the same as those reported by Inthachot et al. [8].

Step 4: Fitness evaluation

Accuracy was employed to guide chromosome selection (subsets of input variables) for generating the next generation in GA, as well as to evaluate the prediction model's performance. The fitness values in GA were defined as the accuracy values, which are calculated using the following formula:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{2.1}$$

Where TP represents true positives, FP is false positives, TN is true negatives, and FN is false negatives.

Step 5: Stopping criterion

Determine whether to continue or exit the loop. The stopping criterion was based on whether the accuracy of the best individual of the current generation was higher than the accuracy of the previous generation. If this continues for 20 consecutive generations (the stall value is 20), ends the program.

Step 6: Selection mechanism

Tournament Selection Process

The selection mechanism in GA ensures that the population of solution candidates consistently improves in terms of overall fitness values. This mechanism allows GA to eliminate sub-optimal solutions while retaining the best individuals. Among the various selection techniques available, Tournament Selection with a size of 3 was adopted in this study for its simplicity, speed, and efficiency. Tournament selection applies higher selection pressure to GA, accelerating convergence and ensuring that the worst solutions do not progress to the next generation.

The tournament selection process involves two main steps. First, players (potential parents) are selected for the tournament. Second, the winner of the tournament is determined based on the highest fitness value. In a size-3 tournament, three chromosomes are randomly selected from the population after removing the elite individuals. The best chromosome among these three, ranked by fitness, is chosen. This process is repeated iteratively until the new population is fully populated.

Step 7: Crossover function

Single-point crossover The crossover operator in the GA combines two parent individuals (chromosomes) to create offspring for the next generation. To perform the crossover operation, two parent chromosomes are selected through the tournament selection process. We implemented the single-point method for the crossover section. This method helps to explore new regions of the search space by exchanging parts of the chromosomes between the parents. The crossover function works by selecting a random crossover point along the chromosome and swapping the genetic material (bits) after this point between the two parents, resulting in two offspring. The benefits of this method are:

- **Promotes Diversity:** By combining different parts of the parent chromosomes, crossover introduces diversity into the population, which helps prevent premature convergence and allows for the exploration of new potential solutions.
- **Maintains Building Blocks:** Single-point crossover ensures that potentially good building blocks (substrings of genes) from both parents are passed onto the offspring, which may lead to the discovery of superior solutions.
- **Faster Convergence:** The crossover process allows the search process to focus on promising regions of the search space, leading to faster convergence towards optimal or near-optimal solutions.

Step 8: Mutation function

Mutation in a genetic algorithm (GA) refers to a genetic perturbation that alters individuals in the population. It plays a crucial role in maintaining genetic diversity and exploring a broader solution space. In this work, we applied uniform mutation as our mutation strategy. The uniform mutation operator randomly selects genes (bits) in a chromosome and flips their values, which introduces variability and prevents the algorithm from getting stuck in local optima by exploring different areas of the solution space. This operator is typically applied with a low probability of preventing drastic changes that could negatively impact the overall fitness of the evolving population. The probability of a mutation occurring in a child's chromosome was set to 10%.

Step 9: Replacement (New population)

The genetic algorithm continues evolving until the new population is fully populated. The new population is formed by adding individuals from Elite kids (There are 2 elite children), Crossover kids, and Mutation kids, as represented by Equation 2.2.

$$\text{New Population} = \text{Elite Kids} + \text{Crossover Kids} + \text{Mutation Kids} \tag{2.2}$$

Once the new population is formed, it is evaluated, and the selection and reproduction processes are repeated until the stopping condition is satisfied.

Step 10: Repeat until occurring the stopping conditions

There are two stopping conditions applicable to this work that are: stopping by hitting the maximum number of generations or when reaching the stall value. The stall value (as mentioned in step 5) and the generation number are 20 and 50 respectively.

Chapter 3

Results and discussion

3.1 Results

Pop_size	ANN	GA + ANN	Best individual
40	52.07%	67.21%	[2 3 5 6 8 10 11 14 15 16 18 19 21 22 23 24 27 36 37 38 39 40]
50	52.07%	68.16%	[0 3 4 5 6 8 11 12 18 20 21 23 24 28 29 34 35 37 38 39 41 42 43]
60	52.07%	68.73%	[1 4 8 9 10 11 16 18 19 20 21 22 25 34 37 40 43]

Table 3.1: Results for different Pop_size values.

The table compares the performance of a genetic algorithm (GA) combined with an artificial neural network (ANN) at different population sizes (40, 50, 60). The standalone ANN achieves a constant accuracy of 52.07%, while the GA + ANN method shows improvements: 67.21% for Pop_size = 40, 68.16% for Pop_size = 50, and 68.73% for Pop_size = 60. This indicates that GA enhances ANN performance by optimizing parameters or feature selection. Larger populations provide more solutions, leading to marginal improvements in accuracy. The best individual features selected by the GA also vary with population size. Overall, GA combined with ANN outperforms the standalone ANN, with the improvements highlighting the benefit of a larger search space. Further analysis of other GA parameters could provide additional insights.

Epoch	ANN	GA + ANN	Best individual
500	52.07%	67.21%	[0 1 3 4 5 8 9 11 12 13 14 16 18 19 20 21 22 24 28 29 32 33 34 36 37 38 39 40 42]
1000	53.88%	68.55%	[0 1 2 3 4 5 8 9 11 12 14 18 19 21 23 25 26 27 30 35 37 41 42 43]
2000	51.34%	67.14%	[0 1 2 3 4 5 8 9 11 12 14 19 21 23 25 26 27 30 35 37 40 42 43]

Table 3.2: Results for different Epoch values.

This table compares the performance of the ANN and GA + ANN methods across different epoch values (500, 1000, and 2000). The results indicate that the ANN method initially shows an accuracy of 52.07% at 500 epochs, which improves slightly to 53.88% at 1000 epochs but then drops to 51.34% at 2000 epochs. This decrease suggests that the model might be overfitting as training progresses beyond a certain point. In contrast, the GA + ANN method performs more consistently, with accuracies of 67.21%, 68.55%, and 67.14% at the respective epochs. The genetic algorithm improves the model's performance and maintains stability over longer training periods. The best individual feature sets evolve with each epoch, showing the GA's capacity to optimize the feature selection for the ANN, with different sets of features being selected across the epochs. Overall, the GA + ANN method consistently outperforms the ANN method, highlighting the advantages of combining genetic algorithms with neural networks for improved performance and robustness.

Mutation Rate	ANN	GA + ANN	Best individual
0.1	52.07%	67.21%	[0 1 2 3 4 5 6 8 9 10 11 12 14 18 20 23 25 26 28 33 34 35 36 37 38 42]
0.2	52.07%	67.21%	[0 1 2 3 4 5 6 8 9 10 11 12 14 18 20 23 25 26 28 33 34 35 36 37 38 42]
0.3	52.07%	69.53%	[0 1 3 5 9 11 16 17 19 20 21 22 23 25 27 29 30 34 35 37 42 43]
0.4	52.07%	68.46%	[0 2 7 8 9 10 11 14 15 19 21 22 25 29 32 34 39 40 41]

Table 3.3: Results for different Mutation Rate values.

The table presents the performance of an artificial neural network (ANN) compared with a genetic algorithm (GA) combined with ANN (GA + ANN) at different mutation rates. The ANN's accuracy remains constant at 52.07% across all mutation rates, indicating that the ANN alone is not optimized by changing mutation rates. However, the GA + ANN method shows improvements in accuracy with higher mutation rates. At a mutation rate of 0.1 and 0.2, the performance remains at 67.21%, but at a mutation rate of 0.3, accuracy increases to 69.53%, showing significant improvement. When the mutation rate is increased further to 0.4, the accuracy slightly drops to 68.46%, suggesting that excessively high mutation rates may disrupt the optimization process. The ”Best individual” feature sets selected by GA change with mutation rates, demonstrating how the GA adapts its search strategy depending on the mutation rate, potentially exploring different combinations of features that contribute to the improved performance. In conclusion, the GA + ANN method consistently outperforms the standalone ANN, with the best performance achieved at a mutation rate of 0.3, highlighting the importance of choosing an optimal mutation rate for effective optimization.

3.2 Discussion

3.2.1 Discussion for pop_size table

The selected features for achieving the best accuracy in the table with a population size (pop_size) include a mix of various technical indicators commonly used in financial analysis. These features, such as WMA (Weighted Moving Average), StochD (Stochastic D), CCI (Commodity Channel Index), ROC (Rate of Change), ADX (Average Directional Index), and RSI (Relative Strength Index), are calculated over different time periods (3, 5, 10, 15). The combination of these indicators allows the model to capture diverse market trends and signals, helping to improve predictive accuracy. This feature set reflects the model's ability to effectively utilize a range of time-sensitive market information to make better-informed decisions.

The features appear frequently because they are calculated over multiple time periods, which allows the model to capture market trends at different time scales. Each technical indicator, such as WMA, StochD, CCI, ROC, ADX, SMA, and WilliamsR, is computed for varying window sizes (3, 5, 10, 15). This approach helps the model understand both short-term fluctuations and long-term trends in the market. Short-term indicators (e.g., WMA_3, SMA_5) reflect immediate price movements, while longer-term indicators (e.g., WMA_15, ADX_15) capture more sustained trends. By incorporating the same feature over different timeframes, the model gains a more holistic view of market behavior, enhancing its predictive accuracy.

3.2.2 Dicussion for epoch table

The selected features for the best accuracy in the epoch table include a combination of different technical indicators across various time periods. These features—such as SMA, WMA, Momentum, Stochastic (StochK, StochD), RSI, MACD, CCI, ROC, and ADX—are widely used in financial analysis to capture trends, momentum, and volatility in market data. The chosen features span multiple time frames, from 3-period indicators (e.g., SMA_3, WMA_3) to longer-term ones (e.g., SMA_15, RSI_15), suggesting that a mix of short-term and long-term information is crucial for achieving high predictive accuracy. The inclusion of various indicators like CCI, RSI, and ROC reflects the need to assess different market conditions to optimize the model's performance.

The features that appear frequently, such as SMA, WMA, Momentum, Stochastic, RSI, CCI, ROC, and ADX across different time periods (3, 5, 10, and 15), are commonly used because they capture essential aspects of financial market behavior. These indicators are designed to analyze price trends, momentum, volatility, and market strength, all of which are crucial for making predictions in financial data. The repeated appearance of these features across various timeframes highlights their robustness in reflecting both short-term and long-term market dynamics. By including them across multiple periods, the model can better account for different market conditions and variations, leading to improved accuracy in forecasting or classification tasks. Furthermore, certain features like RSI, CCI, and Stochastic indicators are known for their ability to identify overbought or oversold conditions, which could significantly enhance the model's predictive capabilities.

3.2.3 Dicussion for mutation table

The features listed above represent the optimal set of technical indicators that contribute to achieving the best accuracy in the mutation rate table. These features include various types of moving averages (SMA, WMA), momentum indicators (StochK, StochD, Momentum), and other technical indicators such as RSI, MACD, CCI, ROC, and ADX, across different timeframes (3, 5, 10, and 15). The selection of these features is based on their ability to capture different aspects of market trends, volatility, and momentum, which are crucial for accurate predictions in the context of mutation rate adjustments. These features consistently appear as significant predictors for the model, highlighting their importance in improving performance and accuracy.

The features such as SMA, WMA, StochK, RSI, MACD, CCI, ROC, and ADX appear frequently because they are key technical indicators that provide essential insights into market trends, momentum, and volatility. These indicators help capture different aspects of market behavior, such as trend direction, overbought or oversold conditions, and price deviations. Their repeated inclusion suggests they are highly relevant for achieving accurate predictions, as they offer complementary information that enhances the model's ability to identify market patterns and improve its performance.

Chapter 4

Conclusion

Investors can utilize features like SMA_3 and SMA_5 to track short-term price trends, while WMA_3 offers a more responsive view by giving weight to recent prices. ROC_3 helps assess stock momentum, signaling whether a stock is gaining or losing strength. Additionally, CCI_5 can identify overbought or oversold conditions, pointing to potential trend reversals. By integrating these indicators, investors can make more informed predictions about stock index movements, improving their decision-making in the market.

In our project, we developed a hybrid model combining Artificial Neural Networks (ANN) and Genetic Algorithms (GA) to predict stock index movements, testing it on a large dataset of historical stock trading data. The goal was to achieve better prediction accuracy compared to a standalone ANN model. The test results demonstrated that the hybrid model successfully met this objective, with an average improvement of 15.14%, resulting in a prediction accuracy of 67.94%. However, the accuracy remains relatively modest, and we are exploring the integration of ANN with other machine learning models to further enhance prediction performance.

Bibliography

[1] F. Ahmad, N. A. Mat-Isa, Z. Hussain, R. Boudville, and M. K. Osman. Genetic algorithm-artificial neural network (ga-ann) hybrid intelligence for cancer diagnosis. In *Proceedings of the 2nd International Conference on Computational Intelligence, Communication Systems and Networks (CICSYN '10)*, pages 78–83, 2010.

[2] E. Altay and M. H. Satman. Stock market forecasting: artificial neural network and linear regression comparison in an emerging market, 2005. SSRN Scholarly Paper ID 893741, Social Science Research Network.

[3] G. S. Atsalakis and K. P. Valavanis. Surveying stock market forecasting techniques—part ii: soft computing methods. In *Expert Systems with Applications*, volume 36, page 5932–5941, 2009.

[4] P.-C. Chang, D.-D. Wang, and C.-L. Zhou. A novel model by evolving partially connected neural network for stock price trend forecasting. *Expert Systems with Applications*, 39(1):611–620, 2012.

[5] A.-S. Chen, M. T. Leung, and H. Daouk. Application of neural networks to an emerging financial market: forecasting and trading the taiwan stock index. In *Computers and Operations Research*, volume 30, page 903–923, 2003.

[6] E. Guresen, G. Kayakutlu, and T. U. Daim. Using artificial neural network models in stock market index prediction. In *Expert Systems with Applications*, volume 38, page 10389–10397, 2011.

[7] J. H. Holland. *Adaptation in Natural and Artificial Systems: an Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. University of Michigan Press, Ann Arbor, Mich, USA, 1975.

[8] M. Inthachot, V. Boonjing, and S. Intakosum. Predicting set50 index trend using artificial neural network and support vector machine. In M. Ali, Y. S. Kwon, C.-H. Lee, J. Kim, and Y. Kim, editors, *Current Approaches in Applied Artificial Intelligence*, pages 404–414. Springer, Berlin, Germany, 2015.

[9] M. Inthachot, Boonjing V, and S. Intakosum. Artificial neural network and genetic algorithm hybrid intelligence for predicting thai stock price index trend. *Computational Intelligence and Neuroscience*, 2016:Article ID 3045254, 8 pages, 2016.

[10] Y. Kara, M. Acar Boyacioglu, and O. K. Baykan. Predicting direction of stock price index movement using artificial neural networks and support vector machines: the sample of the istanbul stock exchange. In *Expert Systems with Applications*, volume 38, page 5311–5319, 2011.

[11] H. Karimi and F. Yousefi. Application of artificial neural network-genetic algorithm (ann-ga) to correlation of density in nanofluids. *Fluid Phase Equilibria*, 336:79–83, 2012.

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

[13] K.-J. Kim. Artificial neural networks with evolutionary instance selection for financial forecasting. *Expert Systems with Applications*, 30(3):519–526, 2006.

[14] T. Kimoto, K. Asakawa, M. Yoda, and M. Takeoka. Stock market prediction system with modular neural networks. In *Proceedings of the 1990 International Joint Conference on Neural Networks (IJCNN '90)*, volume 1, page 1–6, Washington, DC, USA, June 1990.

[15] M.T. Leung, H. Daouk, and A.-S. Chen. Forecasting stock indices: a comparison of classification and level estimation models. In *International Journal of Forecasting*, volume 16, page 173–190, 2000.

[16] W. S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 5(4):115–133, 1943.

[17] J. Patel, S. Shah, P. Thakkar, and K. Kotecha. Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques. *Expert Systems with Applications*, 42(1):259–268, 2015.

[18] T. Phuoc, P. T. K. Anh, P. H. Tam, et al. Applying machine learning algorithms to predict the stock price trend in the stock market – the case of vietnam. *Humanities and Social Sciences Communications*, 11:393, 2024.

[19] K. S. Sangwan, S. Saxena, and G. Kant. Optimization of machining parameters to minimize surface roughness using integrated ann-ga approach. In *Proceedings of the 22nd CIRP Conference on Life Cycle Engineering (LCE '15)*, volume 29, pages 305–310, Sydney, Australia, April 2015.

[20] J. Tian and M. Gao. Network intrusion detection method based on high speed and precise genetic algorithm neural network. In *Proceedings of the International Conference on Networks Security, Wireless Communications and Trusted Computing (NSWCTC'09)*, volume 2, pages 619–622, April 2009.

[21] D. Whitley, T. Starkweather, and C. Bogart. Genetic algorithms and neural networks: optimizing connections and connectivity. *Parallel Computing*, 14(3):347–361, 1990.

[22] S.-I. WU and R.-P. Lu. Stock market prediction of sp 500 via combination of improved bco approach and bp neural network. In *Proceedings of the 21st Annual ACM Computer Science Conference*, pages 257–264, NewYork, NY, USA, Feb 1993.