An Empirical Examination on Forecasting VN30 Short-Term Uptrend Stocks using LSTM along with the Ichimoku Cloud Trading Strategy

Final Year Project Final Report

Pham Ngoc Hai, Hoang Trung Hieu

A thesis submitted in part fulfillment of the degree of BSc. (Hons.)in Computer Science with the supervision of Dr. Phan Duy Hung

Bachelor of Computer Science

Hoa Lac campus - FPT University

07 January 2022

Acknowledgments

We would like to express profound gratitude to all the instructors that we had the pleasure to work with. However, our greatest influence was Dr. Phan Duy Hung. We felt blessed to be under his guidance this year so far. As a teacher, he has detailed knowledge and resources regarding many subjects. As a person, he is extremely attentive, caring while remaining strict and principled. As a researcher, he never lacks enthusiasm in the pursuit of knowledge and always strives to get the best out of himself. In our point of view, we could not ask for a better mentor.

We are also thankful to FPT University for providing the facilities, culture, and opportunities. This institution has greatly nurtured our education, mental fortitude, and ethics.

Abstract

Stock market forecasting is a highly difficult time-series problem due to its extreme volatility and dynamic. This paper proposes a Long-short term memory (LSTM) model that predicts the probability to outperform the market of all of the VN30-Index constituents by using historical price changes along with features calculated from the Ichimoku Cloud trading strategy. After acquiring the pro-posed model's outputs, we buy three stocks with the best probability to sell them ten days later. We then reinvest the money on the next day using the same strategy. The yearly returns of the above trading scheme are used as the empirical results. This study is conducted in a period of 9 years – from the VN30-Index's establishment in 2012 to the end of 2020. On average, the adoption of the Ichimoku Cloud features in the LSTM model makes our trading strategy go from an annual loss of 2.86% to a profit of 14.29%.

Keywords: VN-index, Stock Market, Short-term uptrend forecasting, Ichimoku Cloud, Neural Networks.

Table of Contents

List of Figures

List of Tables

1 Introduction

1.1 Background

Precise prediction of the stock market may generate an enormous amount of profit. As a result, this topic piques a great deal of interest from both financial and scientific researchers. At first, fundamental analysis and technical analysis are the most prominent methods for predicting the market's movements. According to Investopedia [1]:

> *Fundamental analysis is a method of measuring a security's intrinsic value by examining related economic and financial factors. Fundamental analysts study anything that can affect the security's value, from macroeconomic factors such as the state of the economy and industry conditions to microeconomic factors like the effectiveness of the company's management.*

Fundamental analysis has found its footing in the financial world since the dawn of the stock market. This method is also employed by top investors such as Benjamin Graham, Warren Buffet, Micheal Burry and widely considered as the most reliable way to generate long-term returns. Nonetheless, it does have certain limitations. Firstly, fundamental analysts generally use public data from third-party site to compare the targeted stock with its competitors. Therefore, they are prone to false, miscalculated data. Secondly, it doesn't take investors' current consensus into account so there is a notable short-term risk. Thirdly, it has become a somewhat rigid methodology. Traditional valuation formulas used by fundamental analysts are unable value newly emerged intangible assets namely computer software, licences and cryptocurrency. Due to this disadvantages, fundamental analysts have missed out on huge amount of profit technology \

movements based on historical data and calculations. Obviously, this type of analysis doesn't consider a company's financial health so it is vulnerable to long-term risks. Furthermore, since drawings (trendlines and such) and pattern recognition are prevalent, it is extremely susceptible to investors' personal bias. Moreover, there are many traders that tend to use too many technical indicators at once, putting themselves at the risk of overthinking and second-guessing. On top of everything, its linear nature makes it unsuitable for financial time series' nonstationary, nonlinear and high-noise traits [2]. Due to such disadvantages, nowadays, researchers have started to move on from said methodologies and explore more sophisticated approaches that might be more compatible with the chaotic stock market, namely machine and deep-learning architectures and they have found promising results.

1.2 Literature Review

There are many different applications of various models and architectures in order to predict the stock market. One of the most common approaches is using historical prices as inputs then generating either the actual price or price trends (up or down) as outputs. Huynh et al. [3] achieve nearly 60% and over 65% accuracy for the price trends of Standard & Poor's (S&P) 500 index and individual stocks respectively. Moreover, Krauss et al. [4] create a profitable trading strategy by combining deep neural networks, gradient-boosted trees, and random forests to predict stocks' trends. They attain an average daily return of 0.45 percent prior to transaction costs. Ghosh et al. [5] employ Krauss et al. [4]'s framework with improvisation in terms of features and they are able to reach an average daily return of 0.64% using LSTM networks. On the other hand, some researchers try to tackle this topic from unorthodox angles such as Makrehchi et al. [6]. They use labelled social media text as inputs and the trading strategies based on their model are able to outperform S&P 500 index by 20%. In addition, Oncharoen et al. [7] introduce a risk and reward function in their loss function and their proposed model is more effective than traditional trading strategies. Zhang et al. [8] proposed deep and wide area network (DWNN), a new type of neural network that employ a combination of convolutional neural network (CNN) and recurrent neural network (RNN). Their proposed model reduces the margin error in ordinary RNN model by 30%. In 2017, Zhao et al. [9] propose a time-weighted function LSTM model and their empirical evidence shows that it outperforms other well-known models at the time. In Vietnam, researchers have also started to show interest in using deep-learning to predict the local stock market [10-13]. We can observe that LSTM is one of the outstanding neural networks in the domain of predicting the stock market, as many research had optimistic results.

The Ichimoku Cloud is a relevant signals indicator in the world of trading and its potency has been investigated by researchers. Gurrib and Elshareif [14] construct an automated trading model based on the Ichimoku Cloud in a narrow sector (S&P 1500 Composite Energy Index). Their empirical results indicate that the Ichimoku Cloud is able to yield profit even in a declining market. Compare to the previous study, Lim et al. [15] tap into a more expansive, including stocks from both Japan and the US. They conclude that the Ichimoku Cloud can generate profitable signals across both markets. However, not all studies find that the Ichimoku Cloud is a reliably profitable indicator, as [16] is not able to create a consistently profitable model. Thus, the Ichimoku Cloud remains an understudied signals indicator in the context of forecasting the stock market using deep-neural networks.

1.3 Motivation

Although it is evident that the studies above have created great contribution in this field, there are certain limitations. Firstly, the Ichimoku Cloud, for the most part, is only used as an automated trading strategy. We believe that the Ichimoku Cloud is capable of more than just that. For example, we can use its calculation as features in a neural network. Moreover, most studies conducted in this field use the accuracy as the main criteria for their proposed models. This is especially true in Vietnam's current literature, where researchers are mostly concerned about prediction accuracy instead of practical application. While this is the standard practice for a scientific framework, creating wealth is still the ultimate goal behind predicting the stock market. Therefore, the effectiveness of a prediction model should truly show in a realistic financial setting instead of purely accuracy.

1.4 Objectives and contribution

Our paper's contribution lies in three aspects. Firstly, we demonstrate how the Ichimoku Cloud trading strategy can be implemented in an effective deep-neural network by comparing the performance between our proposed model with features from the Ichimoku Cloud model and the same model but without said features. We also present our own implementation of the Ichimoku Cloud instead of just using its original calculations as features. Secondly, we propose a combination of a model and a practical, profitable trading strategy specialized for a niche stock market (Vietnam). Finally, in this paper we follow a traditional scientific framework while evaluating the performance by the standards of the modern financial world. Hence, our study may prove to be valuable to both scientific researchers and financial experts.

1.5 Organization

The remainder of this paper is organized as follows. In section 2 we expand upon the related theories. Section 3 covers our methodology in elaborated details. Section 4 indicates our empirical results. Lastly, in section 5, we discuss the result and shed some light on what may come in our future works.

2 Related theories

2.1 Recurrent Neural Network

2.1.1 Definition

RNN is a powerful class of artificial network typically uses sequential or time series data, based on the work of Rumelhart's work for modelling time series [16]. A notable distinguishing feature of RNN is that it has "memory" that stores the states of previous inputs to influence the next output of the sequence, whereas an ordinary deep neural network considers inputs and outputs to be independent of each other. In order to fully explain the mechanism of RNN, a working knowledge of a typical feed-forward neural network is necessary. Figure 1 illustrates how information moves in RNN and feed-forward neural networks.

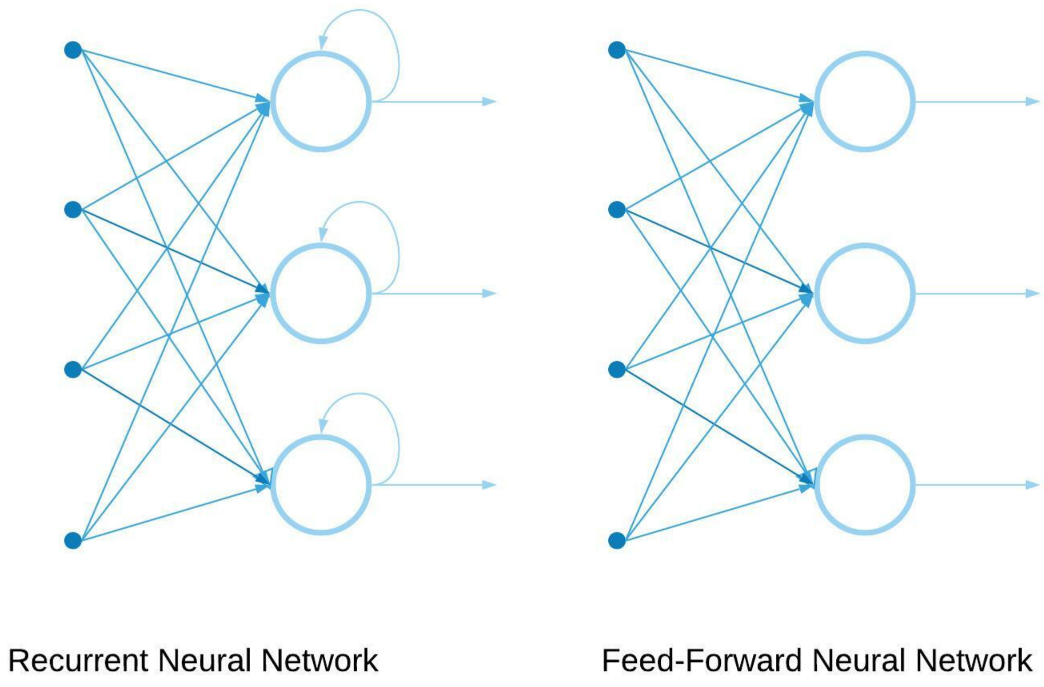

Figure 1. Comparison of RNN and Feed-forward Neural Networks

We refer to this example of the difference in information flow between RNN and feed-forward neural network for a detailed explanation [18].

2.1.2 Types

Figure 2 indicates different types of RNN

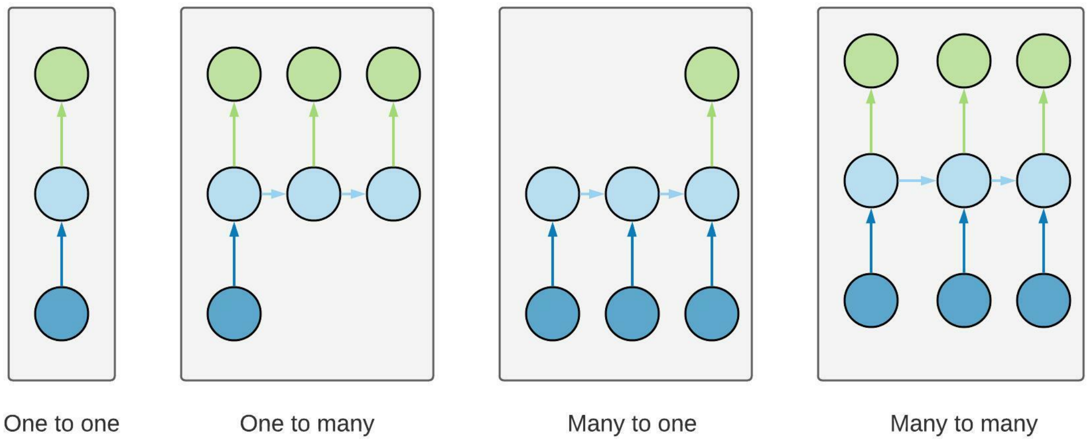

Figure 2. Types of RNN

2.1.3 Limitations

There are 2 major obstacles to the performance of RNN: exploding gradient and vanishing gradient.

2.1.3.1 Exploding gradients

Exploding gradients happen when large error gradients accumulate and result in very large updates to neural network model weights during training. This issue typically can be solved by truncating or squashing the gradients

2.1.3.2 Vanishing gradients

Vanishing gradients are when the values of gradients are too small and the model either stops or takes a very long time to learn. Unlike exploding gradients which can be easily solved by truncating or squashing the gradients, this was a major problem in the 1990s. Later, it was properly addressed with the introduction of LSTM.

2.2 LSTM

2.2.1 Definition

LSTM is “a specific recurrent neural network (RNN) architecture that was designed to model temporal sequences and their long-range dependencies more accurately than conventional RNNs” [19] and it was first brought up by Hochreiter and Schmidhuber to address the problem of vanishing gradient in RNN [20]. Similar to RNN, LSTM also has a hidden state which acts as the short-term memory of the network. Moreover, LSTM adds a cell state which functions as the long-term memory of LSTM. Information added to the cell state or removed from it are controlled by gates. There are three gates in the structure of LSTM: forget gate, input gate and output gate. Figure 3 illustrates the components of a LSTM cell.

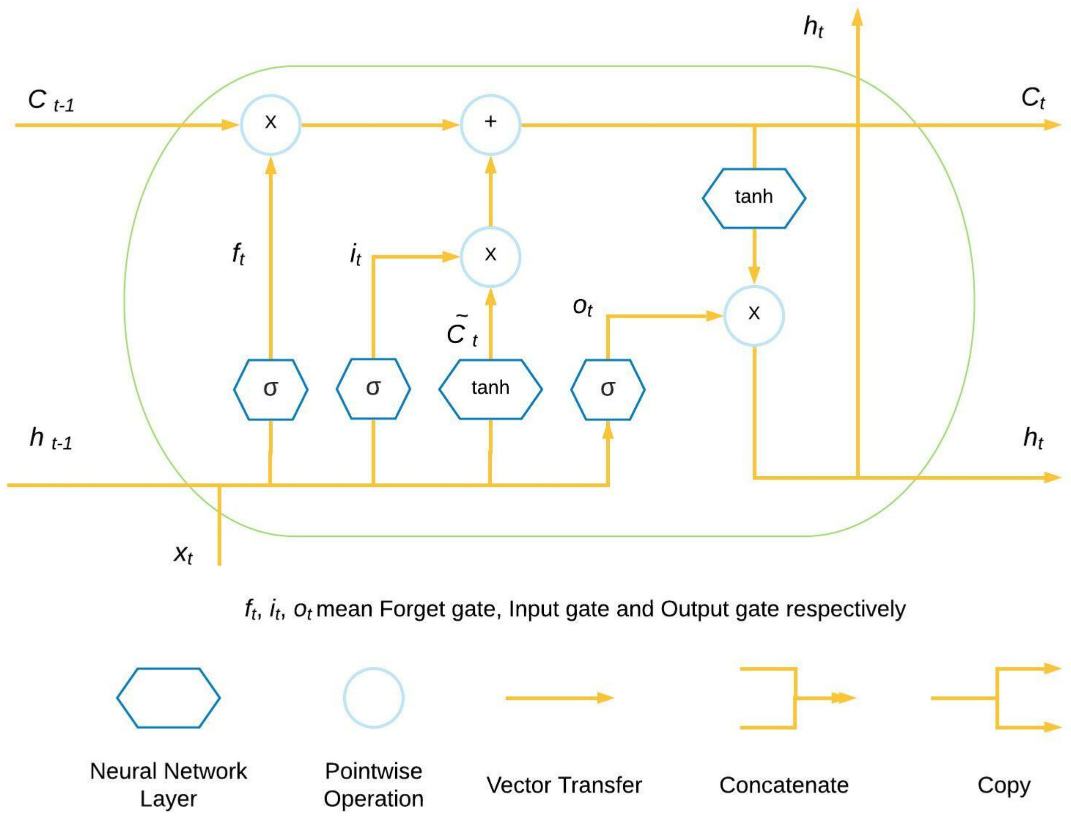

Figure 3. The components of a LSTM cell

2.2.2 Activation functions in LSTM

There are 2 activation functions in used in a LSTM cell: the Sigmoid function and the Tanh activation function

2.2.2.1 Sigmoid function

Figure 4 shows the curve and equation of the Sigmoid function.

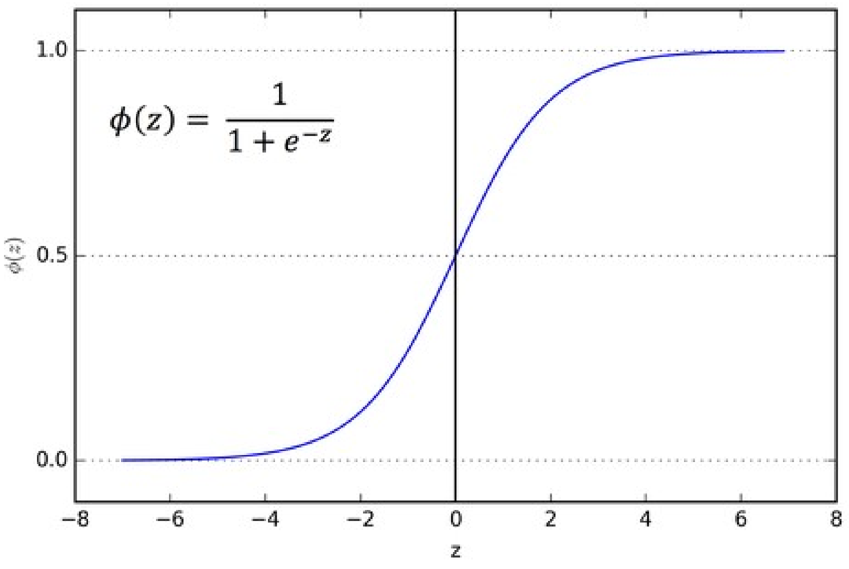

Figure 4. The Sigmoid function [21]

The Sigmoid function is used as the gating function for three gates (Forget, Input and Output) in LSTM because its output is a value between 0 and 1. Only such values can dictate the decisions required in a gate (From forgetting everything to remember everything).

The properties of the sigmoid function:

- It is differentiable.
- It is monotonic unlike its derivative.
- It can cause the neural networks to stuck.

2.2.2.2 Tanh activation function

Figure 5 shows the curve of the Tanh activation function compare to the Sigmoid function.

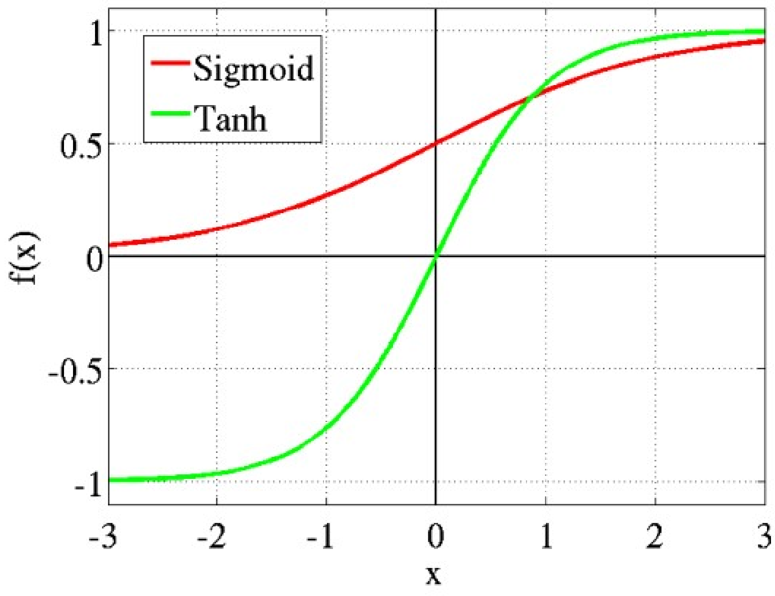

Figure 5. The Tanh activation function [21]

The intuition of using tanh function in LSTM can be explained as of follows. The Sigmoid function is used as the gating function because it can let either no information or complete flow of information pass through the gates. In order to bypass the vanishing gradient issue, a function whose second derivative can sustain for a long range before going to zero. And the tanh function meets all the criteria above.

2.2.3 The mechanism of LSTM

Notation: For clarification purposes, we first explain the notations used in this section's equations

- σ: represents the sigmoid function.
- x_t: the input to the current timestamp.
- U_x: the weight matrix associated with inputs for the respective gates.
- H_t: the hidden state of the current timestamp.
- H_{t-1}: the hidden state of the previous timestamp.
- W_x: the weight matrix associated with the hidden state for the respective gates.
- N_t: vector of new candidate values
- $b_{x:}$ biases for the respective gates

- C_t: cell state at the current timestamp
- C_{t-1}: cell state at the previous timestamp

A LSTM cell takes the previous cell state and hidden state (H_{t-1} and C_{t-1}) as inputs and generate a new cell state and hidden state as outputs. Firstly, LSTM decides whether to keep or throw away information from the previous timestamp (c_{t-1}). This decision is made by the forget gate. We have this equation for the forget gate:

$$f_t = \sigma (x_t * U_f + H_{t-1} * W_f + b_f) \qquad (2.1)$$

The next step is deciding what new information we are going to add to the cell state. Firstly, the input gate decides which values will be updated. Secondly, a tanh layer generates a vector of new candidate values, N_t. These are the equations used in both parts of the second step:

$$i_t = \sigma (x_t * U_i + H_{t-1} * W_i + b_i) \qquad (2.2)$$

$$N_t = \tanh (x_t * U_c + H_{t-1} * W_c + b_c) \qquad (2.3)$$

Next, after obtaining the calculations from both forget gate and input gate, LSTM updates the current cell state using the following equation:

$$C_t = f_t * C_{t-1} + i_t * N_t \qquad (2.4)$$

Finally, LSTM calculates the current hidden state with the following formulas:

$$o_t = \sigma (x_t * U_o + H_{t-1} * W_o + b_o) \qquad (2.5)$$

$$H_t = o_t * \tanh(C_t) \qquad (2.6)$$

2.3 Ichimoku Cloud

2.3.1 Candlestick Chart

A complete understanding of the Ichimoku Cloud requires the knowledge of candlestick chart. A specific period candlestick shows the open, high, low, close price of the corresponding period. The price range between the open and close is considered the real body of the candlestick. If the real body of a candlestick is red, we can infer that the close was lower than the open and the reverse is true when the real body is green. Figure 5 illustrates both types of a candlestick bar.

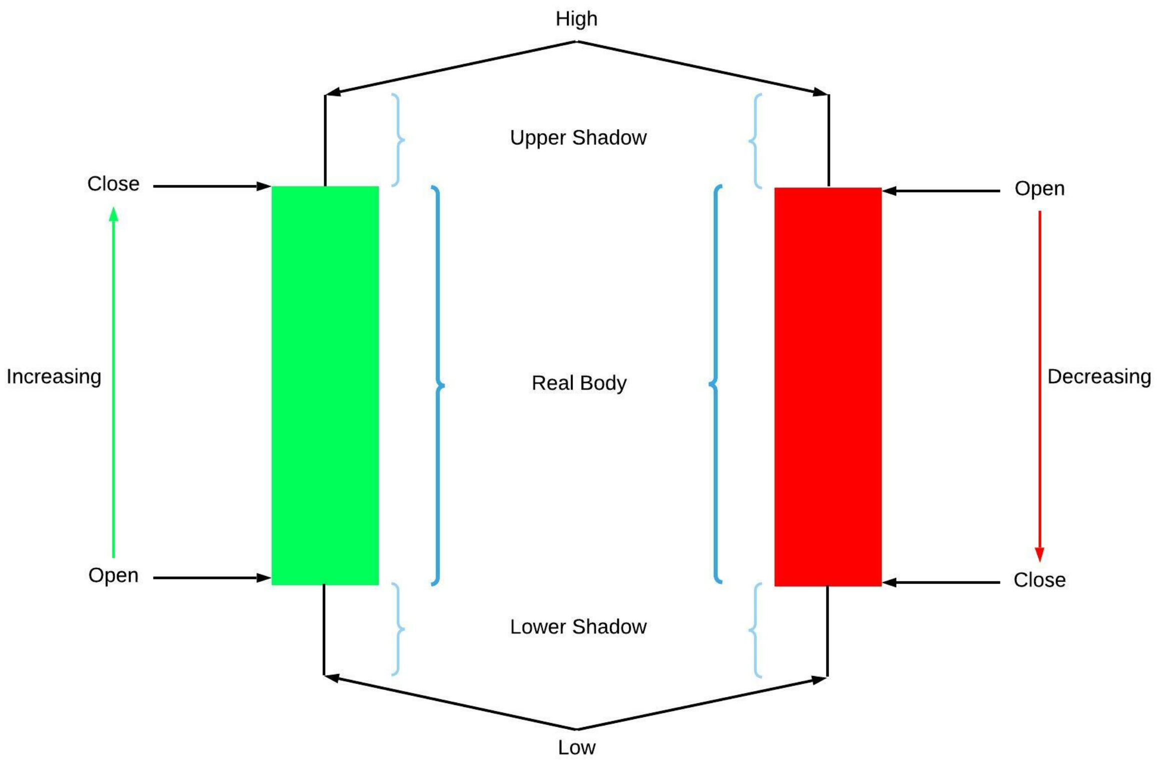

Figure 6. Candlestick Bars

2.3.2 Definition

The Ichimoku Cloud was developed in the late 1930s by Goichi Hoshoda by incorporate moving averages together with candlestick chart. On the one hand, comparing to standard candlestick charts, the Ichimoku Cloud contains more data points and signals. On the other hand, as opposed to other common moving-average based systems, its calculations also use 50% point of highs and lows instead of just closing price.

2.3.3 Key elements

- Conversion line

$$\text{(9-Period High + 9-Period Low) / 2} \tag{2.7}$$

- Base line

$$\text{(26- Period High + 26- Period Low) / 2} \tag{2.8}$$

- Leading Span A

$$\text{(Conversion line + Base line) / 2} \tag{2.9}$$

- Leading Span B

$$(52\text{- Period High} + 52\text{- Period Low}) / 2 \tag{2.10}$$

- Lagging Span: Close plotted 26 periods in the past.

- Cloud: It is the space between Leading Span A and B. The Cloud provides an indicator to the current trend and potential future support, resistance. The market is likely to be bullish (in an upward trend) when the cloud is green and bearish (in a downward trend) when the cloud is red. A cloud's strength is reinforced by its thickness and angle.

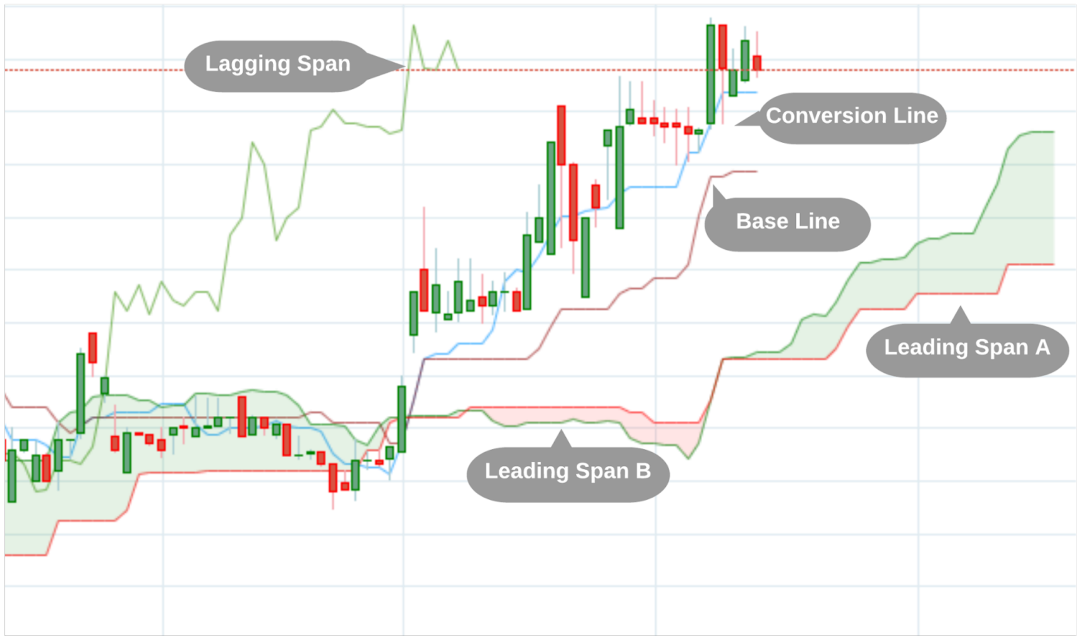

Figure 7. The Ichimoku Cloud

2.3.4 Trading Signals

Buying signals:

- When conversion line crosses from below to above base line. The strength of this signal depends on where it occurs. If it occurs below the cloud, it is weak. If it occurs inside the cloud, it is neutral. Finally, if it occurs above the cloud, it is strong. Figure 8 illustrates when this signal happens.

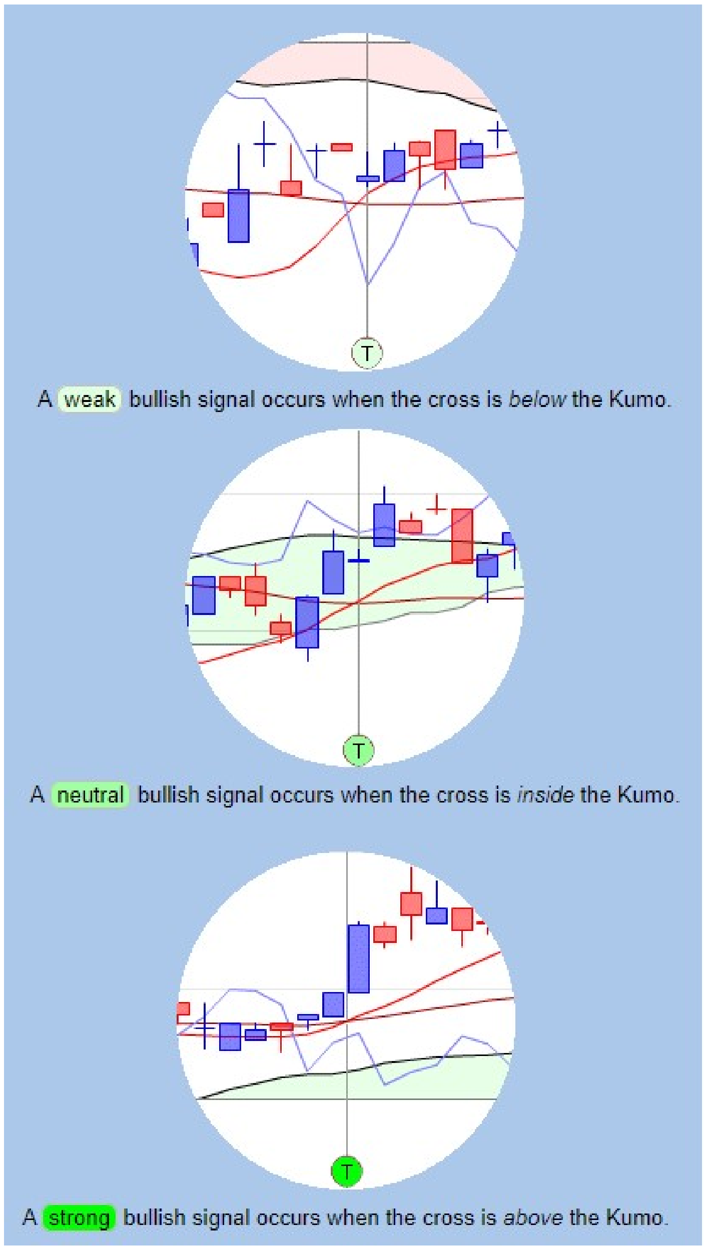

Figure 8. Conversion line / Base line bullish cross

- When real body of a candlestick crosses from below to above baseline. The strength of this signal is measured in the same method as the above signal.

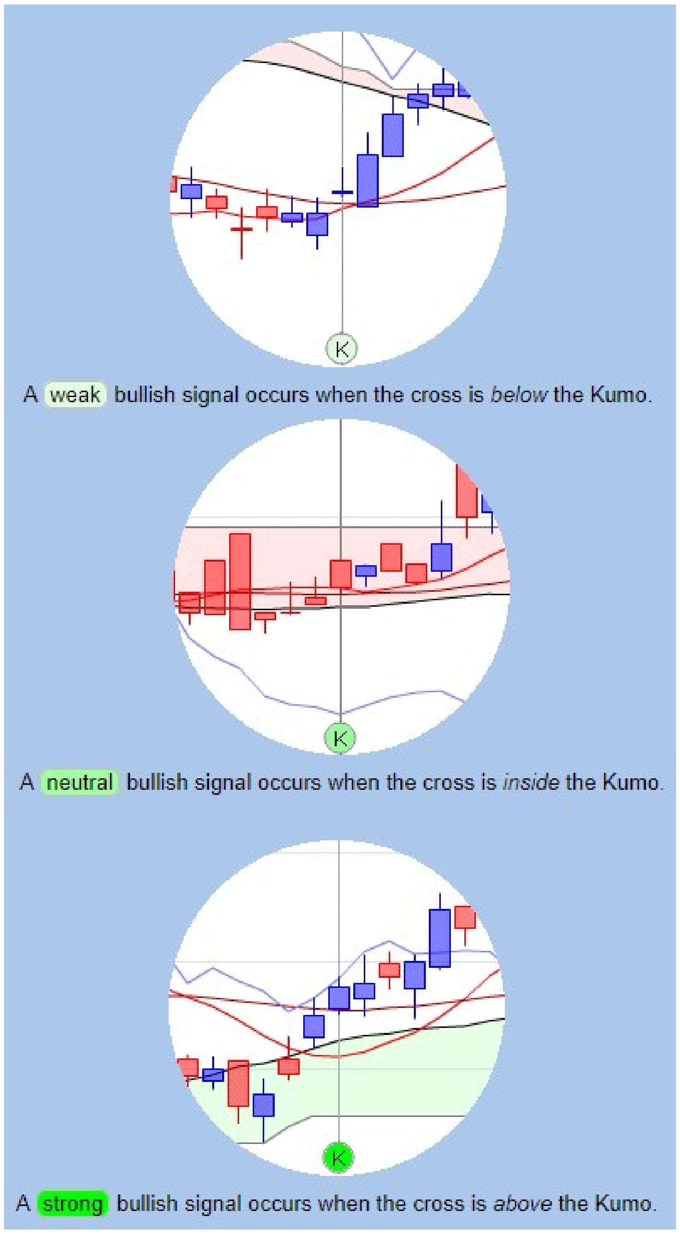

Figure 9. Price / Base line bullish cross

- When leading span A crosses from below to above leading span B. The strength of this signal is measured in the same method as the above signals.

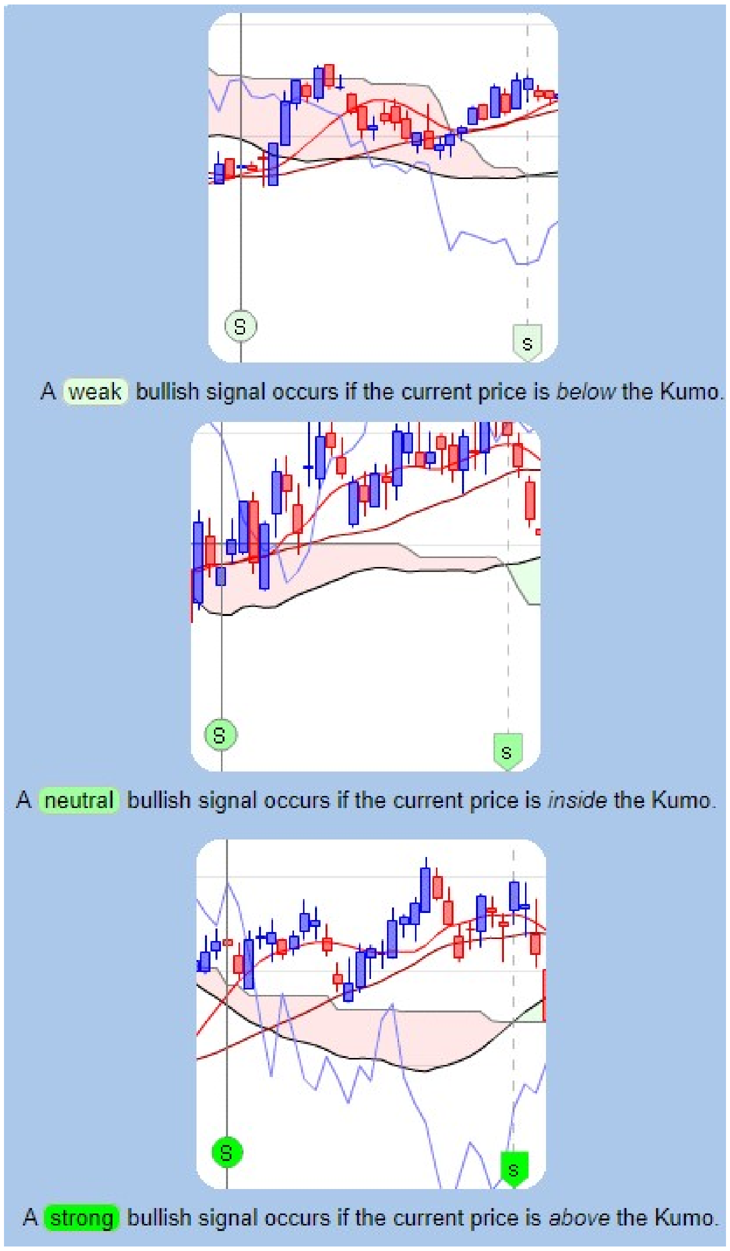

Figure 10. Leading Spans bullish cross

- When the real body of a candlestick crosses from below to above the cloud.

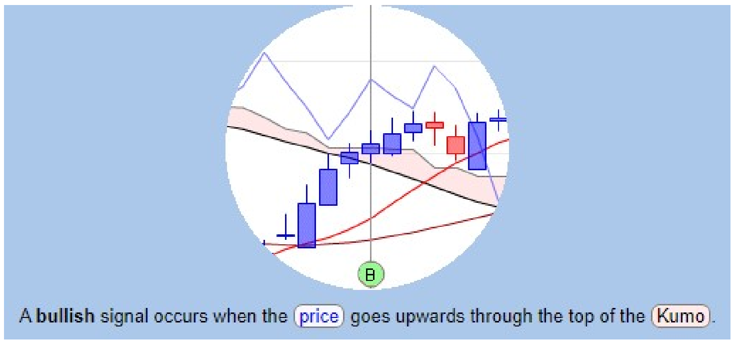

Figure 11.   Bullish Cloud breakout

Selling signals:

- When conversion line crosses from above to below base line. The strength of this signal depends on where it occurs. If it occurs above the cloud, it is weak. If it occurs inside the cloud, it is neutral. Finally, if it occurs below the cloud, it is strong.

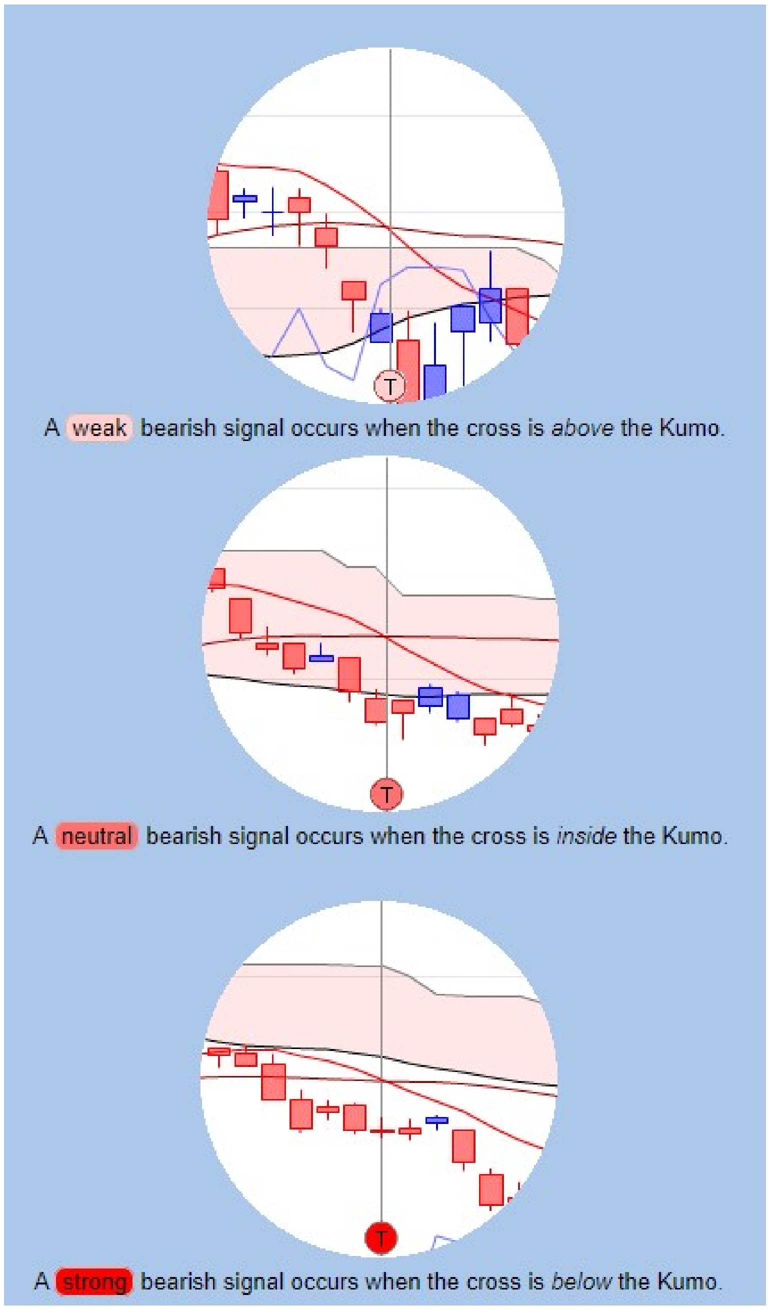

Figure 12. Leading Spans bullish cross

- When real body of a candlestick crosses from above to below baseline. The strength of this signal is measured in the same method as the above signal.

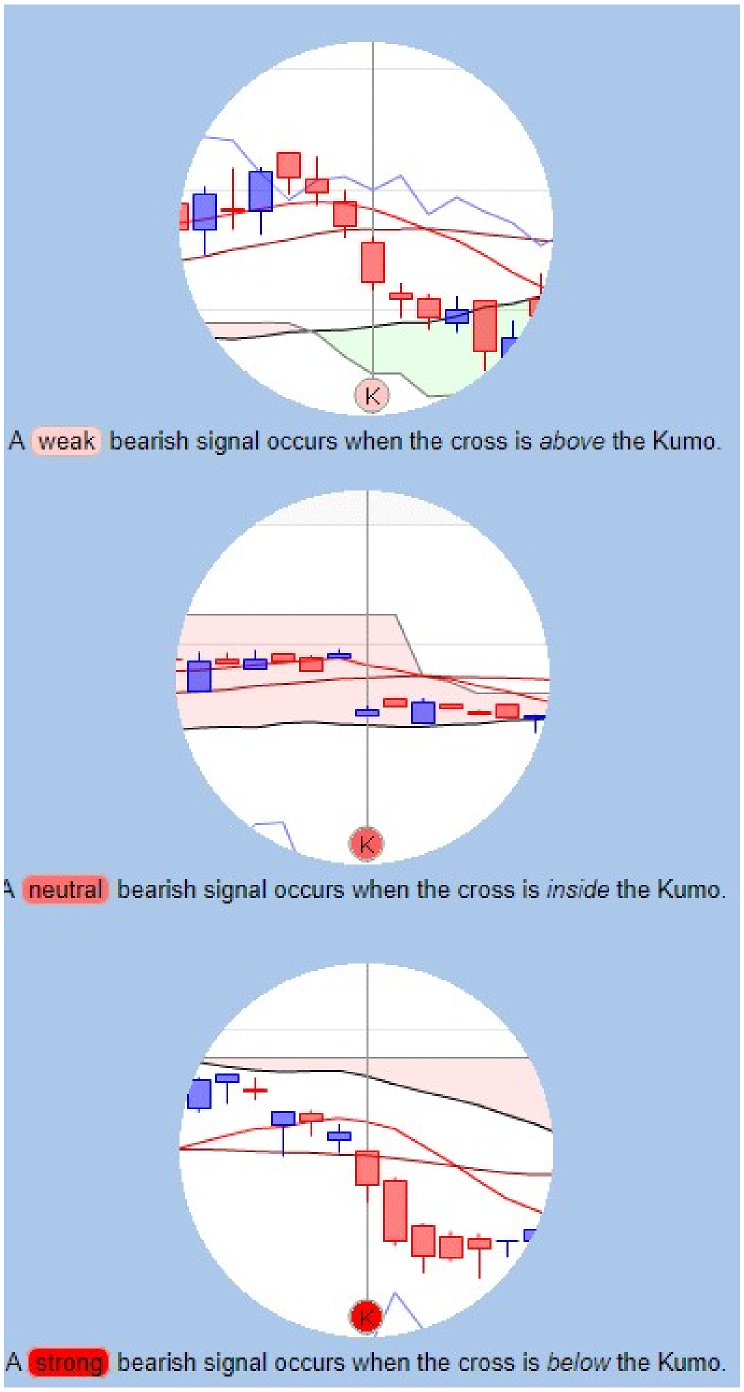

Figure 13. Conversion line / Base line bearish cross

- When leading span A crosses from above to below leading span B. The strength of this signal is measured in the same method as the above signals.

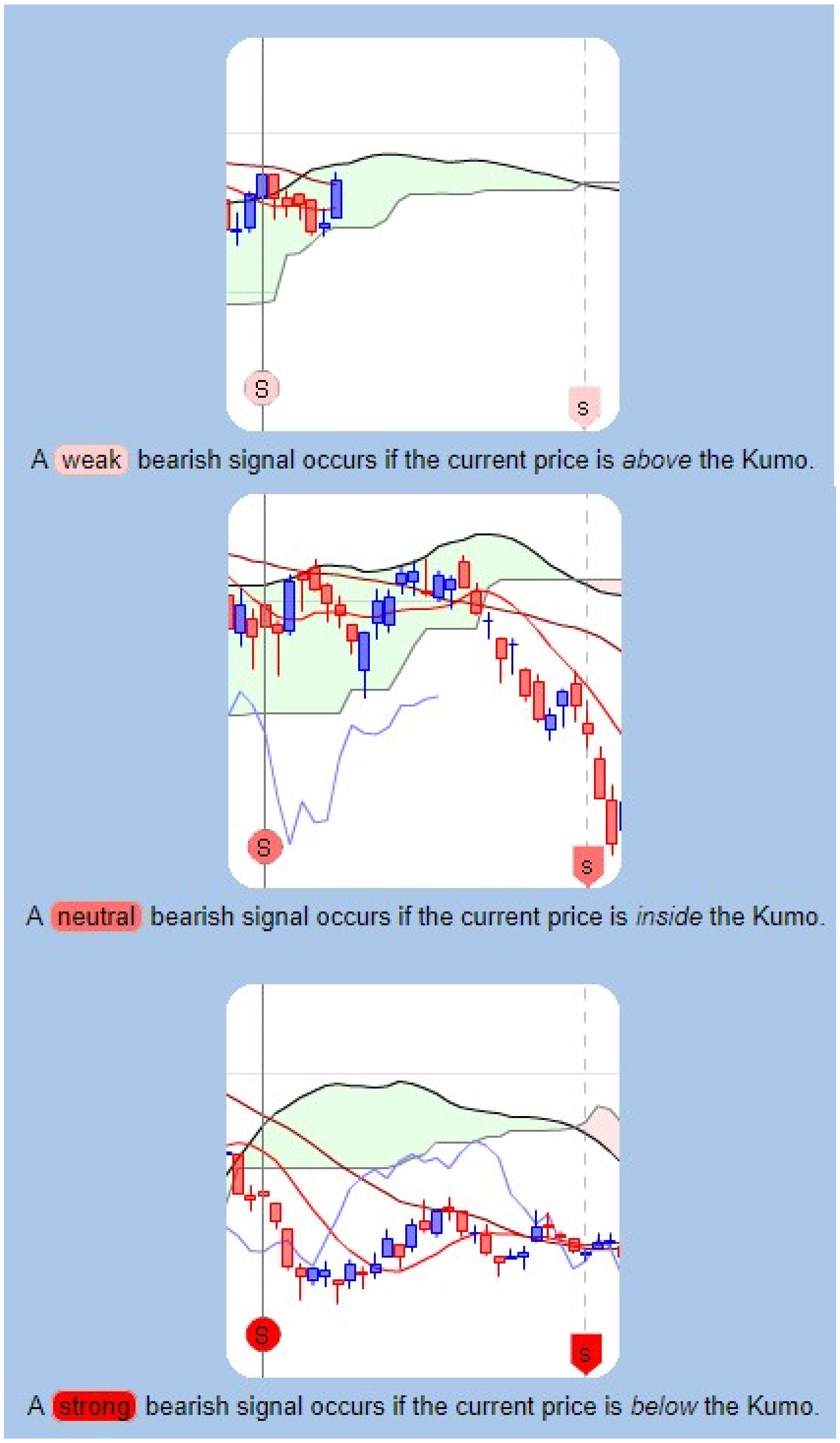

Figure 14. Leading spans bearish cross

- When the real body of a candlestick crosses from above to below the cloud.

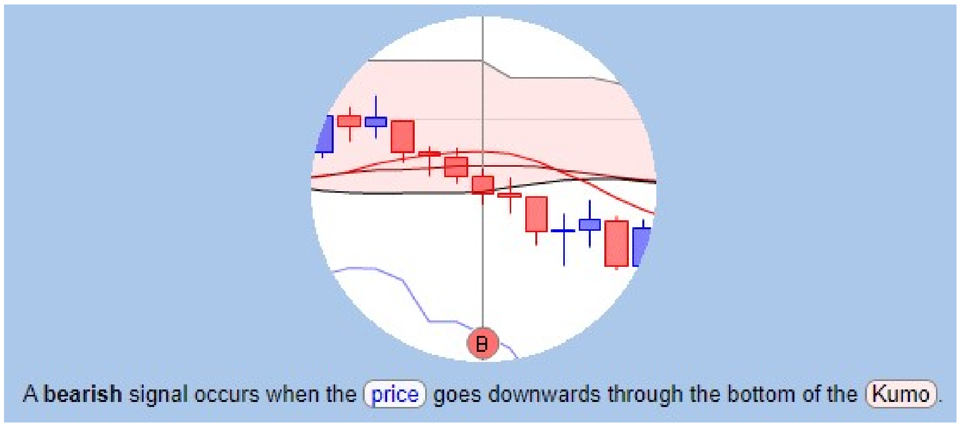

Figure 15. Bearish Kumo Breakout

2.4 Root Mean Square Propagation

Root Mean Square Propogation (RMSProp) has an intriguing background. It was first proposed in lecture 6 of a Coursera online course named “Neural Networks for Machine Learning” by Geoff Hinton. RMSprop is an adaptive learning rate method, which has seen an increasing popularity as well as criticism [22]. It is an unpublished algorithm, yet being famous and well-known as many deep learning researchers use it as part of their frameworks.

2.4.1 Resilient backpropagation

A complete understanding of RMSprop requires a working knowledge of Resilient backpropagation (Rprop). Rprop is a first-order optimization algorithm, first proposed by Riedmiller and Braun in 1992 to address the problem gradients might have wide difference in magnitudes [23]. Determining a single global learning rate is extremely challenging if some gradients are tiny while the others are gigantic. Rprop tries to resolve this issue by only using the sign of the gradient and adapting the step size individually for each weight. There are 4 steps in Rprop:

- Look at the signs of the last two gradients for the weight.
- If they have the same sign, accelerate it by a small margin. If they don’t have the same sign, decrease the step size.
- Limit the step size between two values.
- Apply weight update.

2.4.2 From Rprop to RMSProp

While Rprop does have its advantages, it is not optimal for every situation. For example, it requires large batch size update. When the randomness in stochastic gradient descent is big, the updates are very slow. Furthermore, when the datasets are huge, we have to resort to mini-batch update instead of large-batch update, which is the fundamental idea of Rprop. Therefore, Rprop is not applicable in this case.

They key concept of RMSprop is adopting the use of the sign of gradient from Rprop along with the efficiency of mini-batches update and averaging over mini-batches. The following equations are performed in each update:

$$E[g^2]_t = \beta E[g^2]_{t-1} + (1-\beta)\,(\partial C / \partial w)^2 \tag{2.11}$$

$$w_t = w_{t-1} - \frac{n}{\sqrt{[g2]t}} \frac{\partial C}{\partial w} \tag{2.12}$$

2.5 Categorical cross-entropy

Cross-entropy is a common loss function for optimization in machine learning. In multi-class classification tasks when there are discrete values for the output variable, it is usually referred as categorical cross entropy. The categorical cross-entropy loss function computes the following equation:

$$\text{Loss} = \sum_{i=1}^{size} y_i \cdot \log \hat{y}_i \tag{2.13}$$

where size is the number of scalar values, y_i is the target variable and $\hat{y_i}$ is the ith scalar value in the model output.

2.6 Robust Scaler

Normally, machine learning algorithms perform better when numerical features are scaled to a standard range because different features may have a very wide range of units of measure. For example, in a project about housing prices, we have 2 features: the number of bedrooms in a house and the area of a house. Typically, the number of bedrooms in a house can be counted on the fingers of a person’s hand. In some extreme cases, the number of bedrooms may reach hundreds in luxurious residences they are few and far between. Meanwhile, it is a common occurrence that the area of a house is hundreds meter square and it could get bigger when imperial measure system (feet) is employed. In such situation, the larger value can dominate or skew machine learning algorithms. The consequence is that algorithms spend most of their resources for the large values and ignore the features with smaller values.

Standardization is a popular feature scaling technique which involves calculating the mean and standard deviation of each variable. It then uses these calculated values to scale the variables to have a mean of 0 and standard deviation of 1. The biggest issue with this approach is that standardization is prone to outlier values. With many outlier values in the dataset, standardization can become biased.

To address the presence of these outlier values, robust data scaling was brought up. This method calculates the 25th percentile, 50th percentile (median) and 75th percentile. The values of each feature are adjusted by the following formula:

$$\text{Value} = (\text{Value} - \text{median}) / (75^{\text{th}} \text{ percentile} - 25^{\text{th}} \text{ percentile}) \tag{2.13}$$

3 Proposed procedure

3.1 Data sample, baselines and technology

We specifically choose the VN30-index constituents as the targets of our study because we want to avoid stocks with potential price manipulation. In order to be qualified for VN30, a stock must pass strict requirements in terms of liquidity, market cap, free float ratio, and reputation. Therefore, stocks listed in the VN30-index are more likely to reflect the true demand and supply of the market.

There are 2 categories in our list of baselines. The first one is the market indexes (VN30 and VN-index) and some common, safe investments such as gold, saving accounts (we used the saving interest rate of Agribank, one of the biggest banks in Vietnam), and government treasury bonds. We want to see how our trading strategy fares against the actual market and common, proven investments. We collect gold historical performance from Kitco, Vietnam market indexes from FireAnt, Agribank's saving interest rates from the bank's official documents, and government treasury bond's historical rates from Investing.com. In the second category, we include the performance of our LSTM model using the same trading strategy but without the features from the Ichimoku Cloud to highlight its effectiveness in improving LSTM's prediction power.

We first create a timeline of all stocks listed in the VN30-Index since its establishment in 2012 to the end of 2020 by collecting news from CafeF, Vietnam's latest economic, financial, and securities news channel. We then collect the dividend-adjusted open and closing prices of stocks from the timeline of Cophieu68.vn, a web-site specializing in stock analysis, daily stock market commentary.

The experiments are carried out in Google Colab. We implement our source codes with the assistance of the following libraries: Pandas, Numpy, Tensorflow, and Warnings.

3.2 Methodology

This study follows the general procedure from Krauss et al. [2]. There are 4 major phases. In the first phase, we split our data samples into study periods and then divide our study periods into training and test sets. In the second phase, we select the inputs and outputs necessary for our model. The third step is establishing the setup for our model. Ultimately, we apply our trading strategy based on the predictions.

3.2.1 Creating training and testing sets

Firstly, we divide our original 9-year (2012-2020) period into "study periods" according to Krauss et al. [2]. The total length of a study period is four years. Each study period is furtherly divided into a training and test set. Since our LSTM model has a time sequence of approximately 1 year (240 days), the first year in a study period is only used to generate features. The remaining three years are split into a 2-year training set and a 1-year testing set. Fig.1 illustrates how we split our dataset.

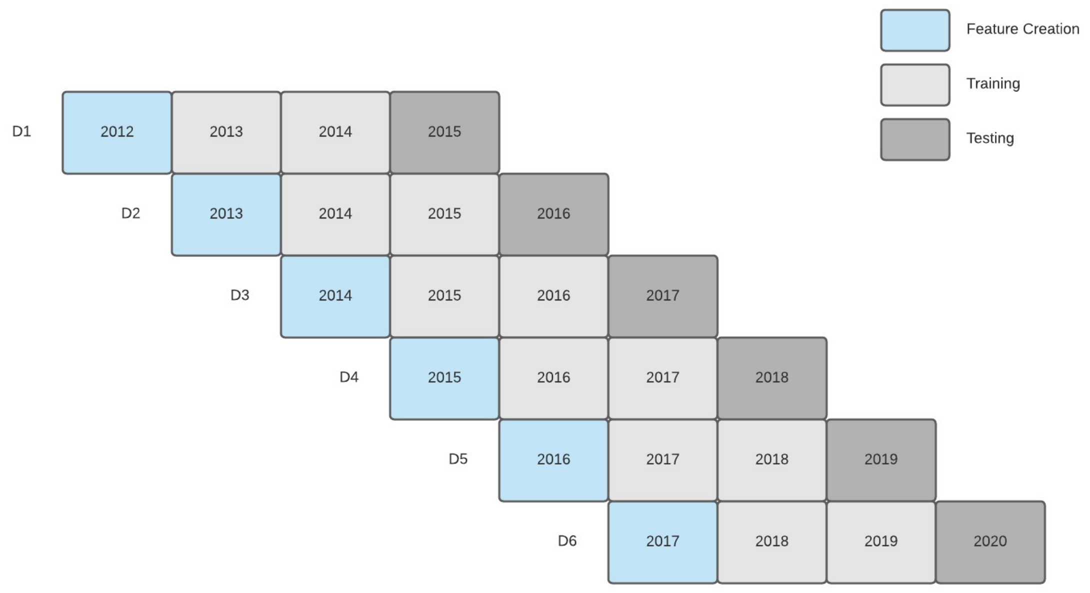

Figure 16. Dataset division

3.2.2 Features and target variable selection

Features selection

Firstly, we introduce the calculations used in the Ichimoku Cloud trading strategy in order to avoid confusion in our further explanation.

- Conversion line

$$(9PH + 9PL) / 2 \tag{3.1}$$

- Base line

$$(26PH + 26PL) / 2 \tag{3.2}$$

- Leading Span A

$$(\text{Conversion line} + \text{Base line}) / 2 \tag{3.3}$$

- Leading Span B

$$(52PH + 52PL) / 2 \tag{3.4}$$

where n-PH stands for the highest closing price in the most recent n days and n-PL is the lowest closing price in the most recent n days.

Our input consists of 240 timesteps and there are 5 accompanying features with each timestep:

- Returns in terms of 10th last closing price:

$$\text{Current Closing Price} / \text{10th Last Day Closing Price} - 1 \tag{3.5}$$

- Four ratios derived from the Ichimoku Cloud trading strategy:

$$\text{Conversion line} / \text{Base line} \tag{3.6}$$

$$\text{Conversion line} / \text{Closing price} \tag{3.7}$$

$$\text{Leading Span A} / \text{Leading Span B} \tag{3.8}$$

$$\text{Leading Span A} / \text{Closing Price} \tag{3.9}$$

We choose to use (3.6), (3.7), (3.8), (3.9) as input features instead of (3.1), (3.2), (3.3), (3.4) because buying and selling signals in the Ichimoku trading strategy focuses on the relationship between the closing price, the lines and spans rather than the actual numbers. Finally, we use Robust Scaler to standardize our features.

Target variable selection

First, we define the cross-sectional median at time t+10 trading return. Perc = [0.5, 0.5] is used for cumulative summing to get thresholds. Then, we use the command pandas.qcut(Quantile-based discretization function) with the 3 parameters:

- x.rank: rank of all percentage change after 10 days
- perc: thresholds for creating label range
- labels=False: Return label 0 and 1

to split the dataset into 2 discrete bins. The result is a column with values of 1 or 0, representing class 1 (if the corresponding stock return after 10 days is bigger than the cross-sectional median value of all stocks at time t) and class 0 (if the corresponding stock return after 10 days is smaller than the cross-sectional median value of all stocks at time t). Using these labels as target variables, our model predicts the probability for each stock in VN30 to outperform the cross-sectional median return in period t+10.

3.2.3 Model specification

Our model has 2 layers of 25 LSTM cells each that precede a dropout layer of 0.1 and a dense layer with 2 output nodes.

- Loss function: we use categorical cross-entropy

- Optimization: we use RMSprop
- Batch size: 64 with epochs=200
- Early stop: patience of 10 epochs, monitoring the validation loss
- Validation ratio: 0.2

3.2.4 Trading strategy

Our model predicts the probability for each stock in VN30 to outperform the median return after ten days. Then, we bought the shares of three stocks that have the highest chance then sell it ten days later. We then reinvested the recouped money in the following day using the same scheme. In order to generate more samples, we split the initial principle into three smaller investments. Each investment was spent on a different day from the others. Figure 9 illustrates how our strategy works.

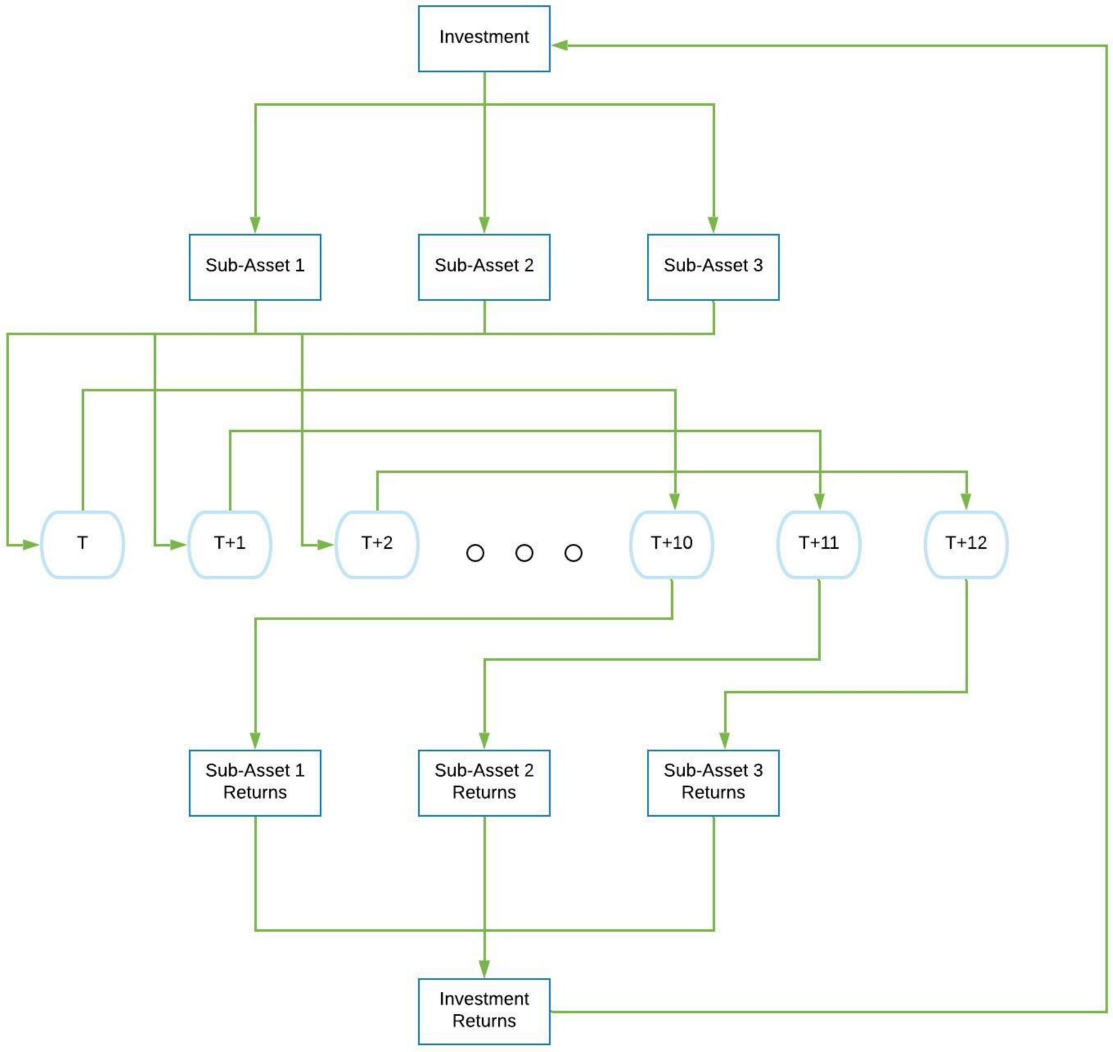

Figure 17. Trading strategy

In this trading strategy, our original principle was 100 000 000 VND. For the ease of calculations, we assume that we can spend approximately the same amount of money on each stock and we can use up all of the available money to purchase new stock. Every purchase was deducted by 0.1% to compensate for brokerage fee and the sale money was deducted by 0.2% (Both brokerage fee and tax account for 0.1% each).

4 Experimental Results

4.1 Training performance

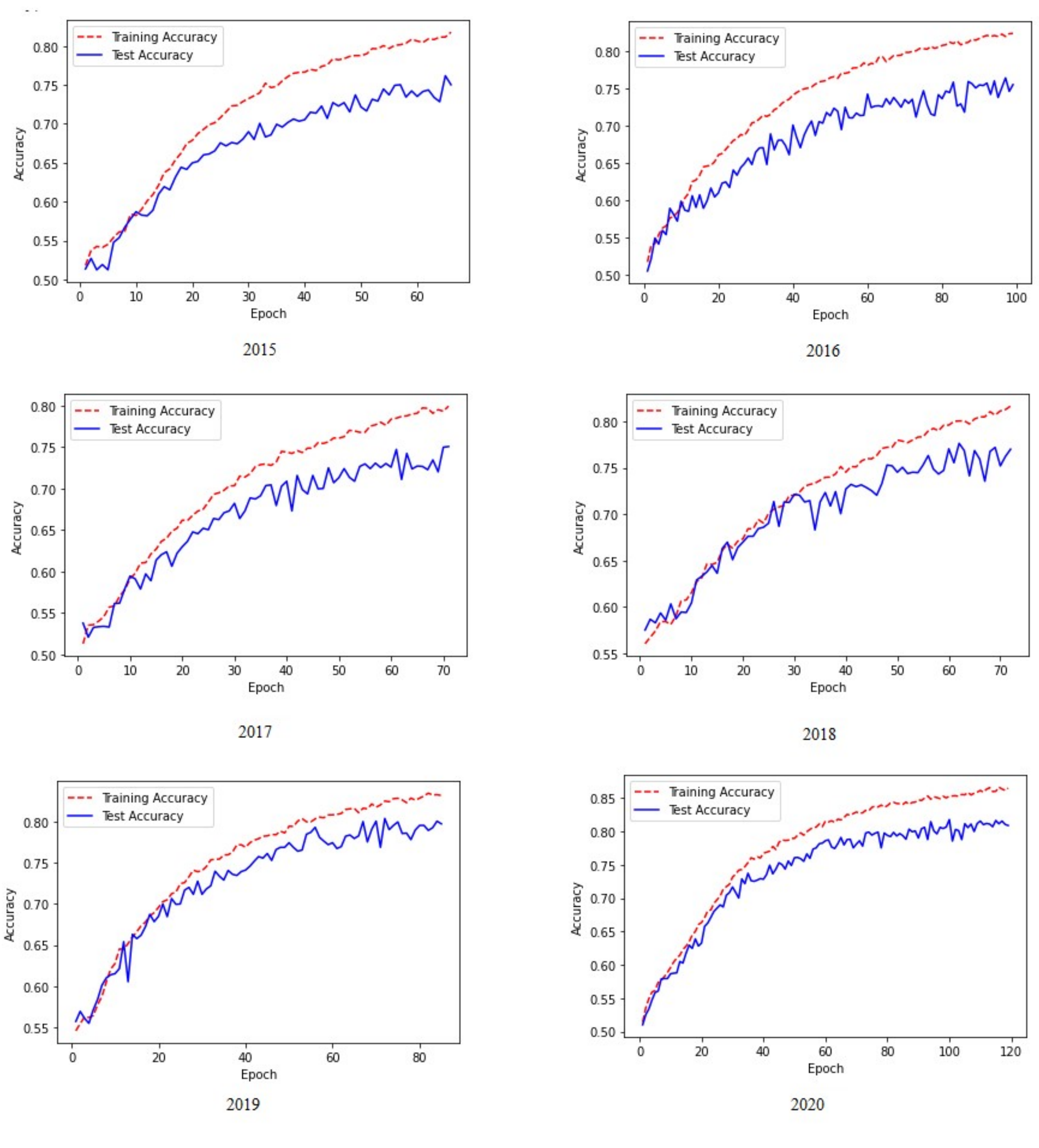

Figure 18. Our proposed model's training performance

Figure 10 illustrates our proposed model training performance. Initially, accuracy hovers around 50% percent. Early stop happens when training accuracy is over 80% and test accuracy is just below 80%. Then, at the time of 70-120 epoch, a divergence occurred as training accuracy keeps going up while test accuracy showed no sign of significant improvement.

4.2 Financial performance

Firstly, we compare the financial performance of our proposed model to the performance of the very same model but without the Ichimoku Cloud features. The intent is to highlight the effectiveness of the Ichimoku Cloud in terms of improving prediction power. Table 1 indicates the how both models performed when applied together with our trading strategy in the test set.

TABLE I. COMPARISONS OF THE PERFORMANCE WITH AND WITHOUT THE ICHIMOKU CLOUD.

	With Ichimoku				**Without Ichimoku**			
Year	Sub-investment 1	Sub-investment 2	Sub-investment 3	Average	Sub-investment 1	Sub-investment 2	Sub-investment 3	Average
2015	+33.21%	+16.35%)	+46.0%	+31.849%	-22.59%	-30.24%	-10.89%	-21.239%
2016	+17.24%	+6.89%	+22.12%	+15.418%	-19.96%	-18.3%	+3.14	- 11.708%
2017	+21.41%	+27.46%	+16.74%	+21.87%	+24.85%	+58.4%	+62.07%	+ 48.436%
2018	-23.56%	-22.86%	+4.78%	-13.878%	-19.42%	-27.83%	-12.9%	-20.049%
2019	+10.56%	+22.31%	+4.11%	+12.328%	+8.65%	+5.82%	-0.88%)	+4.529%
2020	+25.12%	+14.32%	+14.97%	+18.134%	-2.72%	-25.34%	-23.44%	-17.168%

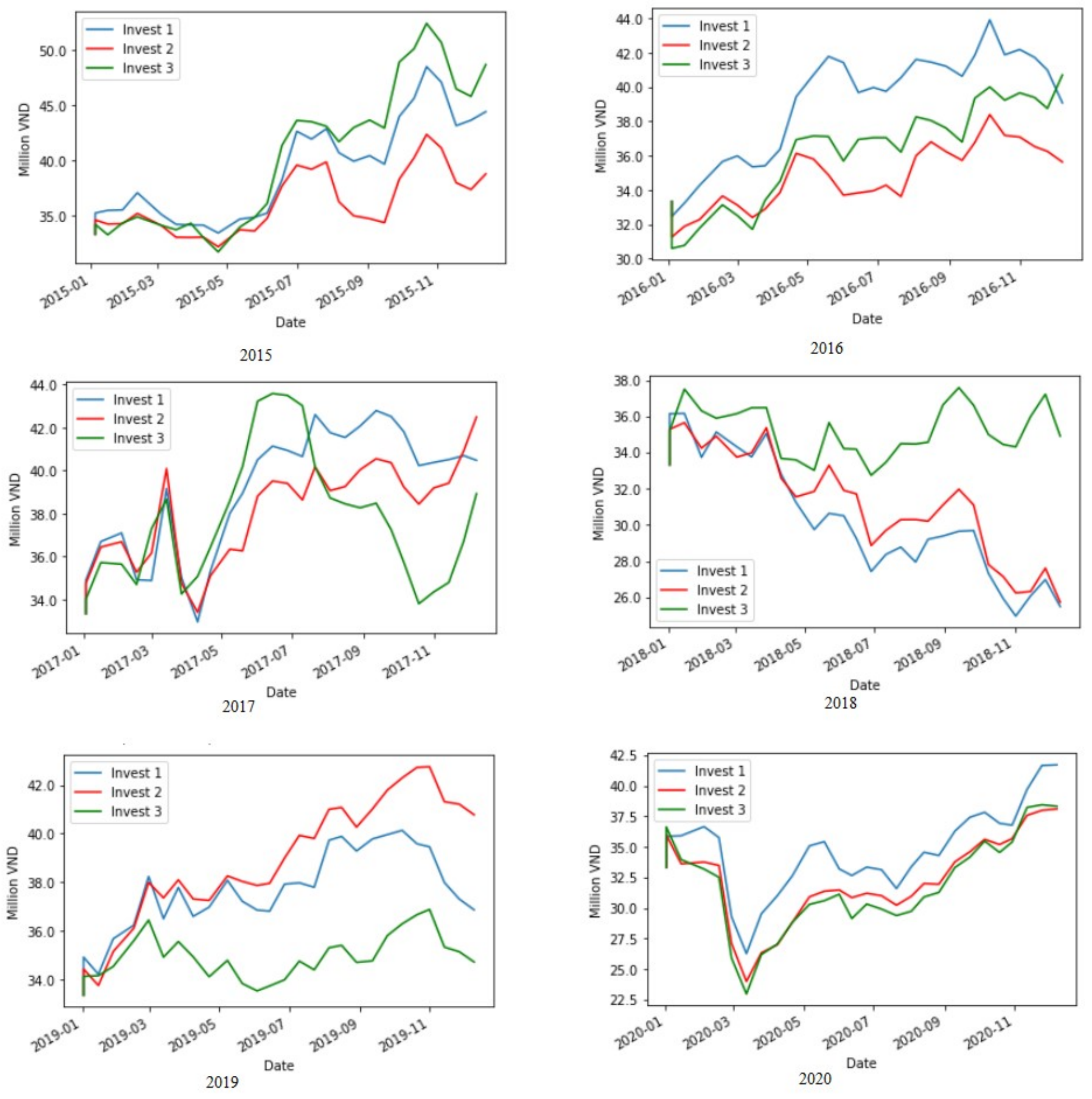

Figure 19. Cumulative money growth during the study periods (Proposed Model)

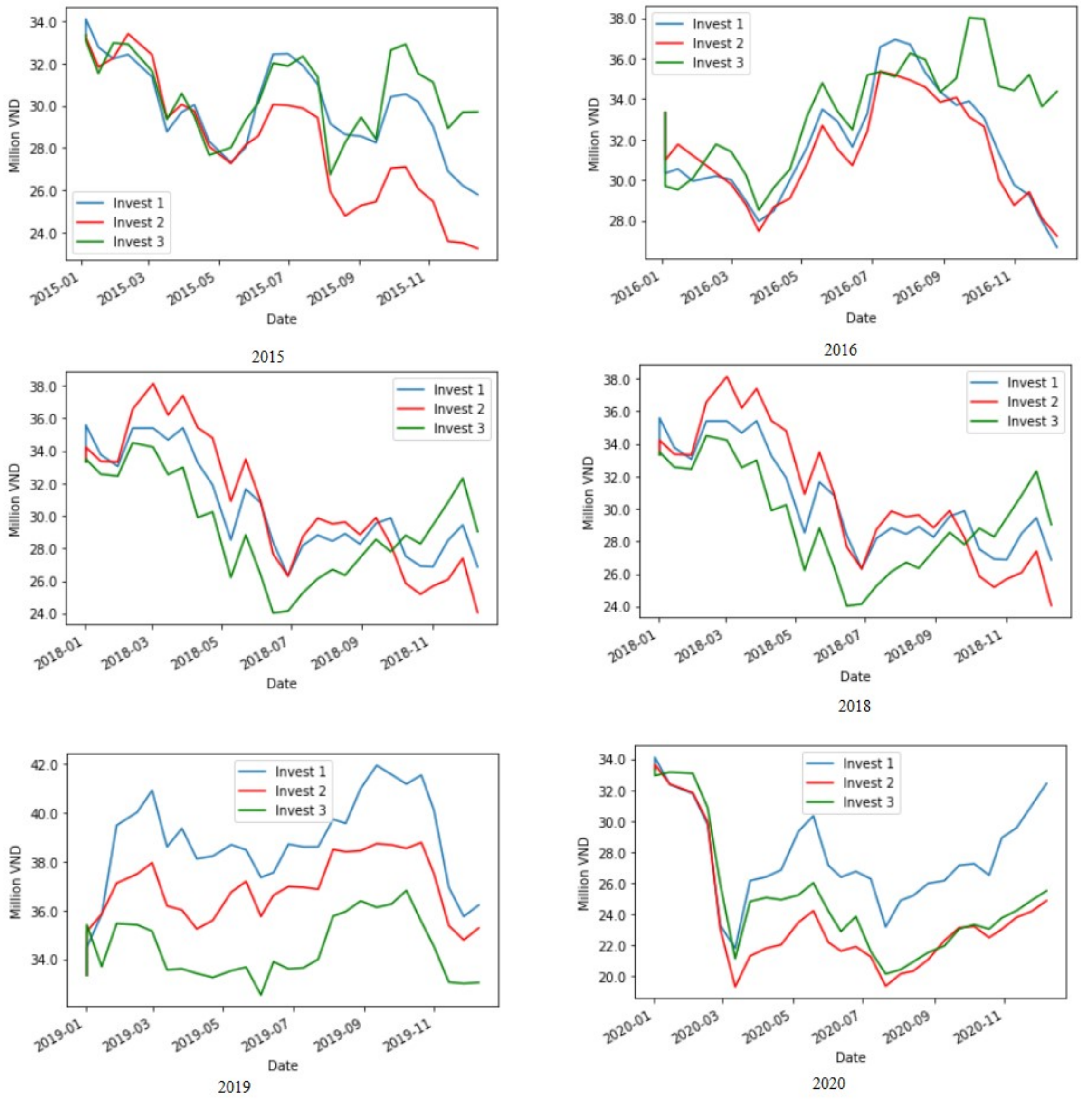

Figure 20. Cumulative money growth during the study periods (Proposed Model without features from the Ichimoku Cloud)

Overall, the implementation of the Ichimoku Cloud features greatly enhances the performance of LSTM. Our proposed model with the Ichimoku Cloud features outperforms its non-Ichimoku counterpart in 5 out of 6 years (2017 being the exception) by a fair margin. Moreover, the non-Ichimoku suffers losses in 4 out of 6 years while our proposed model only sees a loss of 13.878% in 2018. On average, the annual gains of our proposed model with the Ichimoku Cloud (14.287%) dwarfs the non-Ichimoku Cloud version (-2.867%). It even turned what is originally a notable loss to a significant profit. The empirical evidence taken from this table is capable of proving that features extracted from the Ichimoku Cloud indeed improves the prediction power of LSTM.

Table 2 indicates the financial performance of our models in comparisons with real market indexes and safe investments.

TABLE II. THE FINANCIAL PERFORMANCE OF OUR MODELS IN COMPARISIONS WITH REAL MARKET INDEXES AND SAFE INVESTMENTS.

	This paper's strategy	**VN30-Index**	**VN-Index**	**Vietnam 1-year saving**	**Gold**	**Vietnam 10-year Treasury Bond**
2015	+31.849%	-1.01%	+6.12%	+6.2%	-11.59%	+6.43%
2016	+15.418%	+5.48%	+14.82%	+6.5%	+8.63%	+7.03%
2017	+21.87%	+55.29%	+48.03%	+6.5%	+12.57%	+6.01%
2018	-13.878%	-2.36%	-9.32%	+6.3%	-1.15%	+4.09%
2019	+12.328%	+2.82%	+7.67%	+6.8%	+18.83%	+4.88%
2020	+18.134%	+21.81%	+14.87%	+4.9%	+24.43%	+3.15%
Average	+14.287%	+12.01%	+13.7%	+6.2%	+8.62%	+5.31%

On the one hand, it can be inferred from Table 2 that our trading scheme's performance is very competitive with the performance of the local stock market indexes. Overall, the annual gain of our methodologies slightly outperforms the growths of the VN30-index and VN-index by approximately 19% and 4.2% respectively. It is also worth mentioning that market indexes are not subjected to brokerage fees and taxes like our strategy. Therefore, being able to outperform the market indexes with such handicaps is a considerable achievement. In terms of consistency, our strategy outperforms the market growths in 3 years (2015, 2016, and 2019) out of 5 years.

On the other hand, on average, our trading strategy outperforms all other safer and more common investments during the 9-year study timeframe.

5 Conclusion

The technology world is ever changing and evolving. Advancements in modern technology have improved every aspect of our life but the people are still demanding for more. Automated stocks trading using deep-neural networks fits nicely into this trend, since the stock market can be a steady source of income, yet it requires constant monitoring of both macro-level and micro-level economic situations. Especially in Vietnam, applying deep-learning in the stock market is a virtually unexplored domain since the 15-year-old market is still at infantile stage compare to 100-year-old giants such as Dow Jones.

In order to contribute to this topic, this paper proposes a LSTM model that predicts the probability of a VN30 stock outperforming the market and an accompanied trading strategy with an annual return of 14.247%. Outperforming the market indexes is an important requirement to a successful trading strategy and we accomplish just that. We also aim to showcase the Ichimoku Cloud as a viable technical indicator that can be implemented as features in deep-learning models. Our empirical resulted showed that by using features from the Ichimoku Cloud, the performance of our proposed model heavily improved, turning a loss into a substantial profit. Besides, our study follows the general procedures of a scientific research while using financial standards as criteria for empirical analysis. Thus, experts from both academic and financial worlds may find this research valuable.

There are several prospects in our future works. Firstly, the trading strategy employed in this paper is somewhat pristine. In the future, we may consult with professional financial advisors to construct more sophisticated trading principles in our scheme. Lastly, the architecture used in this LSTM model is fairly basic. The first step towards improving our LSTM model might be adding more layers or cells. Another possible direction is employing a more advanced version of LSTM such as stacked LSTM, bidirectional LSTM, and hybrids. In addition, attention mechanism, as indicated in the work of Qiu et al. [24], can improve the performance of LSTM and we may look to incorporate this in the next framework. We can also make a more comprehensive framework by using an ensemble of models instead of just a single model. This approach will reduce generalization error since the components in an ensemble are diverse and work independently from each other.

6 References

1. Segal, T., 2021. What is fundamental analysis? Investopedia. Available at: https://www.investopedia.com/terms/f/fundamentalanalysis.asp [Accessed September 19, 2021].
2. Bontempi, G., Ben Taieb, S. and Le Borgne, Y., 2013. Machine Learning Strategies for Time Series Forecasting. Business Intelligence, pp. 62-77.
3. Huynh, H., Dang, L. and Duong, D., 2017. A New Model for Stock Price Movements Prediction Using Deep Neural Network. Proceedings of the Eighth International Symposium on Information and Communication Technology, pp. 57-62.
4. Krauss, C., Do, X. and Huck, N., 2017. Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P 500. European Journal of Operational Research, 259(2), pp. 689-702.
5. Ghosh, P., Neufeld, A. and Sahoo, J., 2021. Forecasting directional movements of stock prices for intraday trading using LSTM and random forests. Finance Research Letters, p. 102280.
6. Makrehchi, M., Shah, S. and Liao, W., 2013. Stock Prediction Using Event-Based Sentiment Analysis. In: 2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT). pp. 337-342.
7. Oncharoen, P. and Vateekul, P., 2018. Deep Learning Using Risk-Reward Function for Stock Market Prediction. In: CSAI '18: Proceedings of the 2018 2nd International Conference on Computer Science and Artificial Intelligence. New York, NY, USA: Association for Computing Machinery, pp. 556-561.
8. Zhang, R., Yuan, Z. and Shao, X., 2018. A New Combined CNN-RNN Model for Sector Stock Price Analysis. In: 2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC). IEEE, pp.546-551.
9. Zhao, Z., Rao, R., Tu, S. and Shi, J., 2017. Time-Weighted LSTM Model with Redefined Labeling for Stock Trend Prediction. In: 2017 IEEE 29th International Conference on Tools for Artificial Intelligence (ICTAI). IEEE, pp.1210-1217.
10. Tra, N., Tien, H., Dat, N. and Vu, N., 2019. VN-INDEX TREND PREDICTION USING LONG-SHORT TERM MEMORY NEURAL NETWORKS. Journal of Science and Technology: Issue on Information and Communications Technology, 17(12.2), p .61.
11. Do, Q. and Trang, T., 2020. Forecasting Vietnamese stock index: A comparison of hierarchical ANFIS and LSTM. Decision Science Letters, 9, pp. 193-206.
12. Lien Minh, D., Sadeghi-Niaraki, A., Huy, H., Min, K. and Moon, H., 2018. Deep Learning Approach for Short-Term Stock Trends Prediction Based on Two-Stream Gated Recurrent Unit Network. IEEE Access, 6, pp. 55392-55404.
13. Ngoc Hai, P., Manh Tien, N., Trung Hieu, H., Quoc Chung, P., Thanh Son, N., Ngoc Ha, P. and Tung Son, N., 2020. An Empirical Research on the Effectiveness of Different LSTM Architectures on Vietnamese Stock Market. 2020 International Conference on Control, Robotics and Intelligent System, pp. 144–149.
14. Gurrib, I., Kamalov, F. and Elshareif, E., 2020. CAN THE LEADING US ENERGY STOCK PRICES BE PREDICTED USING THE ICHIMOKU CLOUD?. International Journal of Energy Economics and Policy, 11(1), pp. 41-51.
15. Lim, K.J.S., Yanyali, S. and Savidge, J., 2015. Do Ichimoku Cloud Charts Work and Do They Work Better in Japan?. International Federation of Technical Analysts Journal, 2016 Edition, Forthcoming.
16. Deng, S., Yu, H., Wei, C., Yang, T. and Tatsuro, S., 2020. The profitability of Ichimoku Kinkohyo based trading rules in stock markets and FX markets. International Journal of Finance & Economics, 26(4), pp. 5321-5336.
17. Rumelhart, D., Hinton, G. and Williams, R., 1986. Learning representations by back-propagating errors. Nature, 323(6088), pp. 533-536.
18. Donges, N., A guide to RNN: Understanding recurrent neural networks and LSTM Networks. Built In. Available at: https://builtin.com/data-science/recurrent-neural-networks-and-lstm [Accessed October 2, 2021].
19. Sak, H., Senior, A. &amp; Beaufays, F., 2014. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. Interspeech 2014, pp. 338–342.
20. Hochreiter, S. &amp; Schmidhuber, J., 1997. Long short-term memory. Neural Computation, 9(8), pp. 1735–1780.
21. Sharma, S., 2021. Activation functions in neural networks. Medium. Available at: https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6 [Accessed October 3, 2021].
22. Wilson, A., Roelofs, R., Stern, M., Srebro, N. and Recht, B., 2017. The Marginal Value of Adaptive Gradient Methods in Machine Learning. In: NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems. 57 Morehouse Lane, Red Hook, NY, United States: Curran Associates Inc., pp. 4151–4161.
23. Riedmiller, M. and Braun, H., 1993. A direct adaptive method for faster backpropagation learning: the RPROP algorithm. In: IEEE International Conference on Neural Networks. IEEE, pp.586-591 vol.1.

24. Qiu, J., Wang, B. and Zhou, C., 2020. Forecasting stock prices with long-short term memory neural network based on attention mechanism. PLOS ONE, 15(1), p. e0227222.