A Robust Deep Learning Model for Predicting the Trend of Stock Market Prices During Market Crash Periods

Alireza Ghasemieh, Rasha Kashef
Department of Electrical, Computer and Biomedical Engineering
Ryerson University, Toronto, Canada
alireza.ghasemieh@ryerson.ca, rkashef@ryerson.ca

Abstract— **The stock market is one of the most important investment opportunities for small and large investors. Stock market fluctuations provide opportunities and risks for investors. However, some fluctuations are considered as enormous threats for most investors; significantly when the stock market has fallen sharply due to external factors and does not reach its previous point in a long time. For example, at the beginning of 2020, financial market indices, especially the stock market, fell sharply due to the COVID-19 pandemic, and for a long time, the indices did not grow significantly. Many investors suffered huge losses during this period. Although much research has been done in stock market forecasting and very efficient models have been proposed so far, no special effort has been made to build a model resistant to the collapse of financial markets. We propose a Convolutional Neural Network (CNN)-based ensemble model that is highly resilient to the stock market crash, especially at the beginning of the COVID-19 period. The proposed model not only avoids losing money in financial crises but can bring significant returns to investors. Experimental results show that the ensemble CNN models using Gramian Angular Fields (GAF) has greatly improved the resistance of the model in critical market conditions.**

Keywords: **Stock Trend Prediction, Deep Learning, Convolutional Neural Network, Ensemble Modeling.**

I. Introduction

Stock markets are known as one of the most important places to invest globally. Many financial institutions, banks, legal entities and individuals invest large amounts of their assets in these markets. For this reason, the stock market in each country is one of the indicators of economic growth. That is why stock market changes are so important that they result from many factors such as macroeconomic situation, news, corporate financial reports, parallel market conditions, psychological and emotional conditions of investors, legal components and other hidden factors. For this reason, the ability to accurately predict the stock market trend is desirable to investors and researchers. Given the many factors affecting the stock market, machine learning and deep learning methods can help us to improve forecasts. In this regard, a lot of research has been done and is being done. Due to the high complexity of the stock market behaviour, each research covers a specific part of the market and conditions. Most recently, in early 2020, the COVID-19 pandemic caused stock markets to collapse worldwide, with many investing their assets out of the stock market. This phenomenon caused many investors to suffer huge losses. Despite such phenomena in the past, no research model has been proposed that resists stock market crashes and prevents the loss of investors. Therefore in this paper, we propose a model that can withstand falling stock values and even create profitability which has not been addressed in the past. We propose an ensemble model of Gramian Angular Fields Convolutional Neural Network that shows an awe-inspiring ability to control the critical conditions of the stock market. Using such a model can significantly help investors avoid losing their capital in similar critical situations. In this paper, we have made four contributions towards building our robust model 1) we used novel technical indicators (TI) as accurate representatives of the market, 2) we encoded time series into images for better prediction results, 3) we used Gramian Angular Fields (GAF) to encode a group of technical indicators into RGB images that reveal hidden patterns in time series, and 4) we used a CNN ensemble technique to increase the robustness and profitability of our model. The financial evaluations show that the proposed model is robust to the market crash and interestingly profitable.

This paper is organized as follows: Section 2 discusses the existing methodologies and approaches to predict stock market trends. Section 3 presents the proposed model and the data preprocessing methods and learning models. Section 4 discusses the evaluation methodology. Sections 5 presents the experimental evaluation, and section 6 provides conclusions and future directions, respectively.

II. Related work

Various efforts have been made using statistical analysis methods, machine learning methods and deep learning methods. In the meantime, deep learning methods have shown significantly better performance than statistical methods or traditional machine learning methods. Our review only focuses on the most related deep learning methods in this paper. Before reviewing the previous models in detail, it is necessary to have an overview of different approaches that have been taken toward the prediction of the stock market. In general, there are two categories of research on the use of machine learning in the stock market. The first category is models that predict stock prices in the coming day or the coming days. This group of models [1] [2] [3], which are of regression, aim to determine the stock price, regardless of the market trend. The second category is models that predict stock market price trends in the coming days. In these researches [4] [5] [6] [7] [8] [9], regardless of the stock price, they deal with the stock price trends. This type of research is of the classification type. It is divided into two groups: A group that considers ascending and descending trend classes and a group that considers three ascending, descending and no-significant change classes. In the first category, if the market is bullish, investors take a long position, and if the market forecast is bearish, they take a short position. However, in the second category, there will be three modes: short, long or hold. In this approach [10] [11] [12] [13], there are many different methods for determining the class. Some use a threshold to determine the class. This means that if the rate of change of price is more than α percent, it is considered bullish, and if the price decrease is more than α percent, it is considered bearish. Another group uses a moving window with a specific size for a few days to find the maximum and minimum and consider the rest as a hold. Naturally, in determining this classification method, the obtained classes will not be balanced, creating many problems in the model learning. We can categorize the models based on the deep learning architecture. The first category is the studies that use Multi-Layer Perceptron (MLP) as the core component in their model architecture. Shi et al. [6] proposed a model to predict weekly price change turning points by utilizing MLP over 500 stocks and found out it is profitable for 329 of them. Tabar et al. [14] suggested an MLP model using TI to predict sell, buy and hold action. The same method is used by Zhong et al. [15] for daily stock market forecasting with Principal component analysis (PCA) in the preprocessing stage. Sezar et al. [10] also proposed a trading system based on evolutionary optimized technical analysis (TI) using MLP in their architecture. Many researchers used CNN as the core learning component. Hoseinzade et al. [9] proposed CNNpred to predict market trends using TI with 2D-CNN and 3D-CNN architectures. It showed promising performance in a trading simulation. Sezar et al. [13] proposed a time series to image encoding and a CNN network to predict the buy, sell, and hold classes. Wen et al. [16] used a method to simplify noisy-filled financial temporal via rebuilding the sequence in preprocessing stage and used CNN architecture to forecast the market trends. The model was tested over 8 stocks and showed better performance in some stocks. Yan et al. [17] and Gunduz et al. [18] also used pure CNN with feature selection to predict stock market trends. A group of researchers tried CNN with other learning methods like Long-Short Term Memory (LSTM) and Adaboost. Kim et al. [19], Lu et al. [3], Shi et al. [20], Eapen et al. [21], Lu et al. [2], Wu et al. [22] used CNN-LSTM as the core learning component with some variation in preprocessing. The results of this method proved that it is superior to CNN only. Taherkhani et al. [23] added an Adaboost classifier after a CNN to use transfer learning to the next iteration to enhance the CNN model performance to classify images. In this technique, each CNN classifier learns from the previous CNN mistakes and tries to make a correct prediction. We used the idea to build a CNN-Adaboost ensemble model. Barra et al. [4] used time series-to-image encoding to combine four different time resolutions (1 hour, 4 hours, 8 hours, one day) of the open price and make an image to forecast long and short positions. Wang et al. [24] proposed a novel technique to encode time series as images. They use Gramian Angular Fields (GAF) and Markov Transition Fields (MTF) to generate new images. We used this technique with TI to generate images. We can also classify the models by the evaluation method of the models. Many studies [1] [5] [7] [2] [3] that aimed to predict the next day's price or use loss to evaluate the model performance used the MAE, RMSE, R^2 methods commonly used for regression. Therefore, they compared their results with other studies using those benchmarks. The studies in [5] [11] [12] [6] [8] [22], which are of the classification type, have used the parameters of accuracy, precision, recall, F1 score and Kappa.

III. The Robust Market Forecasting Model

Stock market prediction of the next day can be considered as a classification problem which has three classes: Short position that means sell at the beginning of the day and buy and the end of the day, Hold position that means no sell and buy, and Long position that means buy at the beginning of a day and sell it at the end of the day. We propose a robust prediction model, which comprises the following phases, 1) data preprocessing including time-series to image conversion, 2) learning models, and 3) validation. Figure 1 shows the main flow diagram of our model. First, the stock market raw data is preprocessed through two stages: a technical indicator generator module that generates the TIs listed in Table 3, and a time-series to image encoder that uses two encoding methods: *Spreading Horizontally (SH)*, *Gramian Angular Fields (GAF)*. The preprocessed data in the form of images are fed into 20 CNN base learners, and finally, the output of CNNs is used to create ensemble models. We applied some rules to simplify the trading and make it more realistic. First, all the available funds will be traded at the short and long positions. Second, no position will be kept for the next day, and it should be closed by the end of the day even it loses. Third a fixed trade fee is applied to all buys, which is 5$. This trading fee plays a significant role in the profitability of models, which many studies did not counter. The last assumption is, we consider the initial fund as 10k $, and if a model loses all the money, it will stop the trading. The following subsections provide a detailed description of each of these components.

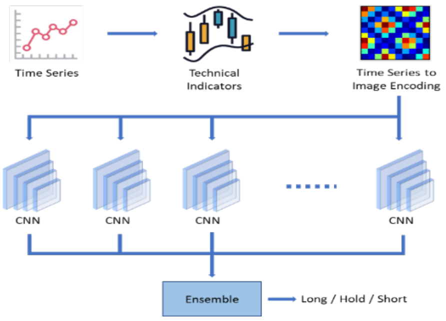


Figure 1 Flow diagram of the proposed model

A. Time Series to Image Conversion

We have adopted two approaches in converting the time-series stock data into an image to obtain better prediction results.

1) Spreading Horizontally (SH)

Time-series or chained inputs can be encoded in a format of an image. The simplest way is to add zeros at the end of the series to adjust the length of the input series so that its length is equal to the square of an integer. Then it fits an image of the size of the square root of the input length. Figure *2* shows an example of this representation method. In this image, regular patterns can be seen. These patterns can be used to categorize images. For example, different patterns can be seen at the bottom of the images, and also at the top of the image, striped lines show the desired patterns. In our study, the size of our sequence is 896, which by adding four zeros, it reaches 900. Then we converted the array into a 2D shape with the height and width of 30 pixels the square of 30. Therefore the images are 30 * 30 pixels.

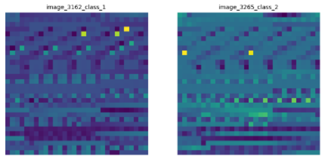


Figure 2 Time-series to image formatting result

2) *Gramian Angular Fields(GAF)*

Another encoding time-series as images is Gramian Angular Fields (GAF) [24], enabling CNN to observe hidden patterns. As Figure *3* shows, X is a sequence of time series. Then X is rescale using Eq.2. The scaled $\tilde{x}$ is transformed into a polar coordinate system using Eq.3. Finally, GAF is calculated as shown in Eq.4. In our case, since the sequence size is 896 using the GAF method, the dimension of the output image is 896* 896 pixels. However, we reduced the resolution to 88*88 to lower the computational cost. Figure *4* shows an example of the output.

$$\tilde{x}_i = \frac{(x_i - \max(X) + (x_i - \min(X))}{\max(X) - (X)} \text{ (1), where } X = \{x_1, x_2, \dots, x_n\}$$

$$\begin{cases} \emptyset = \arccos(\tilde{x}_i)\,,\ -1 \le \tilde{x}_i \le 1\,, \tilde{x}_i \in \tilde{X} \\ r = \frac{t_i}{N},\ t_i \in \mathbb{N} \end{cases} \quad (2)$$

where t_i is the time stamp, and N is a constant parameter to regularize the span.

$$G = \begin{bmatrix} \cos(\emptyset_1 + \emptyset_1) & \cdots & \cos(\emptyset_1 + \emptyset_n) \\ \vdots & \ddots & \vdots \\ \cos(\emptyset_n + \emptyset_1) & \cdots & \cos(\emptyset_n + \emptyset_n) \end{bmatrix} (3)$$

$$= \tilde{X}'.\tilde{X} - \sqrt{I - \widetilde{X^2}'}.\sqrt{I - \widetilde{X^2}} \ (4)$$

where ø is a polar coordinate angle which $\emptyset \in [0, \pi]$

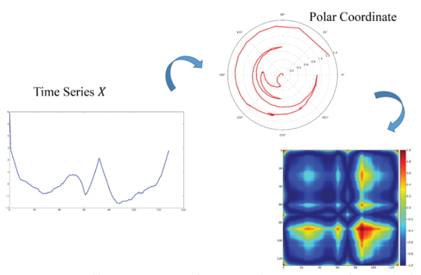


Figure 3 Encoding map of Gramian Angular Field [24]

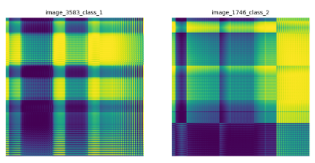


Figure 4 Gramian Angular Fields (GAF) sample output

B. **Learning Models**

In this paper, we used two-dimensional CNNs as the core classifier that extracts the stock market's significant movements from RGB images that represent the market properties. We will discuss this representation later. We used CNN because of its remarkable capacity to extract features hidden in images.

1) *Convolutional Neural Network*

A convolutional neural network is a deep learning method that accepts RGB images as input, extracts significant features in images through multiple filters and updates its neural network's weights to learn. The core function of CNN is discrete convolution operation (5). The first argument (x) is often referred to as the *input,* and the second argument (w) is referred to as the *kernel,* and the output (s) is called the *feature map.*

$$s(t) = (x * w)(t) = \sum \mathrm{x(a)w(t-a)} \quad (5)$$

If the convolution operation applies on two dimensions input (I) like image, it uses a two-dimensional discrete kernel (K) as (6) [25] demonstrates. If the input image is coloured, three channels (RGB) would be applied to convolve the input with the kernel. The kernel size (n*n) is a hyperparameter that should be adjusted. As Figure 5 shows, in order to extract the features in images, multiple filters (feature maps S) should be trained through the convolution operation. To find more complex features or features components additional layer could be added. The convolution process is performed over images with a windowing slide to cover all pixels of images, and the results have a smaller size than the original images. Still, the number of filters will be increased.

$$S(i,j) = (K * I)(i,j) = \sum_m \sum_n I(i-m, j-n) K(m,n) \ (6)$$

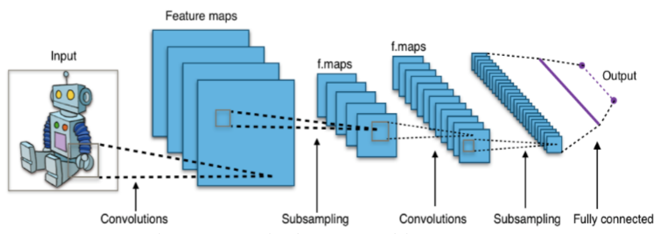


Figure 5 Typical CNN architecture [26]

The base CNN model architecture (Figure 6) has 3 Conv2D networks with 32, 64 and 64 neurons followed by a MaxPool of size two and a dropout at the first layer, another Conv2D with 64, a MaxPool with size two and a dropout as the second layer. The result is flattened; two feed forwards networks with 64 neurons and a dropout enhance the network's performance. Finally, a dense layer with three nodes with a Softmax activation function to predict the class label. ReLU and Softmax are the activation functions for the hidden layer and output layer, respectively.

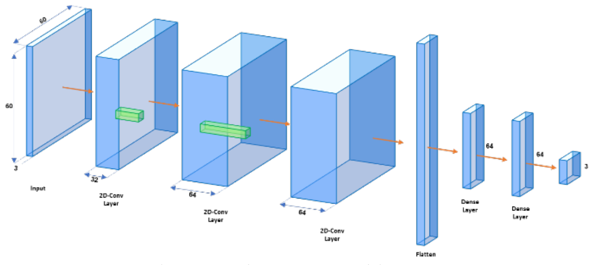


Figure 6 The CNN Architecture

2) *Ensemble Models*

Three well-known ensemble techniques can be used to enhance the performance of the basic model: Stacking, Voting and Adaboost-CNN [23]. In this paper, we developed ensemble models with the same CNN classifier and compared the results with the CNN result and other ensemble models.

Stacking Ensemble Model: The stacking ensemble model (Figure 7) concatenates two-level classifiers: base learner and meta learner. A group of 20 base learners, which are CNN classifiers, learn the model through the training process, and their prediction becomes the input of the meta learner. The meta learner learns which classifier is more reliable in what circumstances and generalizes the forecast to enhance the accuracy of the output. The meta learner is Feed Forward Neural Network (FFNN) with a hidden layer of 20 neurons with ReLU activation function and a dropout of 0.2 followed by a dense layer of 3 neurons.

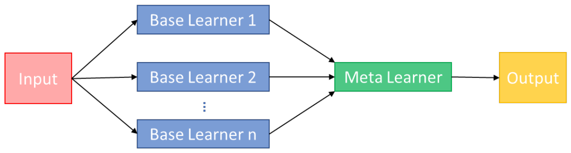


Figure 7 Stacking ensemble model architecture

Voting Ensemble Model: Unlike the stacking ensemble model, the voting ensemble model has one level of classifiers. Still, its decision-making module determines the output based on voting that satisfies a threshold. This threshold is a hyperparameter that, in our case, 0.46 is the best one for all 20 stocks. However, it is possible to have a specific threshold for each stock. Like the stacking ensemble model, we used 20 independent CNN classifiers with the same architecture in Figure 8.

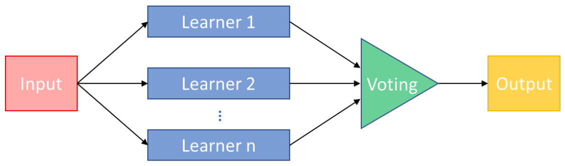


Figure 8 Voting ensemble model architecture

Adaboost-CNN Ensemble Model: Adaboost is a well-known tree-based classifier that uses the transfer learning technique to help a sequence of classifiers learn from the previous classifier mistake and enhance the accuracy by covering all aspects of the sample space. In [23], Taherkhani et al. expanded the Adaboost capability to be applicable over two-dimensional CNN. The empirical results proved that this method could be beneficial for image classification. Figure 9 shows the classifiers that use the CNN transfer learning technique to learn from previous classifier mistakes.

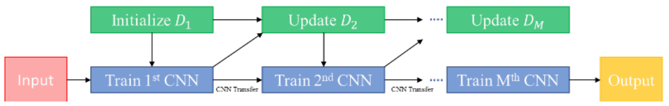


Figure 9 AdaBoost-CNN model architecture

IV. Evaluation Methodology

In this paper, we used two different model evaluation methods: computational and financial evaluation. We used computational metrics to assess the performance of our models. However, we still need to measure the financial performance of the models in the stock market, assuming we have initial capital and we calculate the trading performance as well.

A. *Computational Evaluation*

We use traditional model evaluation metrics to assess the performance of our models: Accuracy [27], Recall [28], Precision, F1-Score and Kappa [29]. Since we considered a trade fee to all buying, either the classifier predicts shot position or long position that each has to pay a 5$. In this situation, the model must avoid *False Positive (FP)* not to cause trading fees. Since we have three classes, *True Positive (TP)* means either the correct prediction of a short or long position and FP means any actual hold position that the classifier considers a short or long position. As mentioned, FP plays a vital role in the profitability of a model. We proposed a cost matrix that shows the logic in Table 1. Generally, if the model predicts TP and *True Negative* (TN), that means it generates profits, and if it predicts FP and FN, it loses money. If it predicts hold as either long or short, it may lose or make money based on market price fluctuation, which does not significantly impact the model's profitability. In all cases, a trade fee (μ) is applied in all buying transactions.

Table 1: Cost matrix

		Predicted Class		
		Long	**Hold**	**Short**
Actual Class	**Long**	P - μ	0	L - μ
	Hold	F - μ	0	F - μ
	Short	L - μ	0	P- μ

P: Profit (+), L: Loss (-), F: Price Fluctuation (+/-)

B. *Financial Evaluation*

Computational metrics are utilized to evaluate the performance of a model. However, the final aim of training a model is its trading capability and the annual return. In this case, we designed some other metrics that show the model's return performance.

1) *Win-Loss Ratio (WLR)*

The win-Loss ratio (Eq.7) [27], [30] is a well-known metric that professional traders use to assess their performance. It is computed by the average winning trade in percentage over the average loss. This metric should be over 1.2 to consider a profitable trading trend

$$WinLossRatio = \frac{\sum Profit\ Percentage/Total\ Win\ Transaction\ Count}{\sum Loss\ Percentage/Total\ Loss\ Transaction\ Count} \quad (7)$$

2) *Batting Average (BA)*

Another crucial statistical tool is the Batting Average [28], the number of winning transactions overall transactions (8). This metric must be over 50% and is suitable if it goes above 60% and is excellent above 65%.

$$Batting\ Average = \frac{Total\ Number\ of\ Win\ Transactions}{Total\ Number\ of\ Transactions} \quad (8)$$

3) *Average Return Per Transactions (ApT)*

The average return is the average profitability of each transaction over one year (Eq.9). Sometimes ApT is high, but the model is not very profitable because of the low number of transactions. It means the model cannot detect trading opportunities well, and it only detects the big spike in the market, not smaller ones.

$$ApT = \frac{Total\ Profit\ in\ Percentage}{Total\ Number\ of\ Transactions} \quad (9)$$

4) *Annual Return (AR)*

Annual return (Eq.10) [29] is the model's profitability during one calendar year. AR is an essential parameter for measuring model efficiency versus other methods like the Buy and Hold strategy.

$$Annual\ Return = \frac{Total\ Asset/Initial\ Balance}{1/Number\ of\ Years} \quad (10)$$

5) *Annual Number of Trades (AT)*

The number of opportunities that a model can detect is important. If a model only detects big spikes in the market, then it means the number of transactions is deficient over a year, and in comparison with other methods, it is not profitable enough [32][33].

6) *Difference from Buy and Hold (BaH) Return (DBnHr)*

A good metric to assess a model performance is comparing its return with the situation that an investor buys a stock, keeps it over a period of time, and gets the gain of the investment. If the proposed model can bit the performance of the BaH strategy, it meets the baseline. However, it should be mentioned that biting the BaH in a very uptrend market is complicated and sometimes impossible. Leaving a long position in a very uptrend market is inefficient that no professional trader does. In this situation bitting, the market trend is almost impossible. In a calm market, the model should overcome the return of the BaH strategy.

V. EXPERIMENTAL EVALUATION

A. *Datasets*

The data is collected through Alphavantage API. It contains 20 stocks as presented in Table 2 with seven attributes of daily records: Ticker, Date, Open Price, High Price, Low Price, Close Price, and Volume.

Table 2: Stock tickers

TICKER	START DATE	END DATE	TICKER	START DATE	END DATE
AAPL	2000-04-20	2021-09-08	LMT	2000-05-23	2021-09-08
AEP	2000-05-12	2021-09-08	MA	2006-10-04	2021-09-08
AFL	2000-04-27	2021-09-08	MO	2000-04-20	2021-09-08
AIV	2000-04-20	2021-09-08	MRK	2000-04-26	2021-09-08
C	2000-04-20	2021-09-08	NEE	2000-04-20	2021-09-08
CL	2000-04-26	2021-09-08	T	2000-04-20	2021-09-08
CSCO	2000-02-18	2021-09-08	USB	2000-04-26	2021-09-08
CVX	2000-04-20	2021-09-08	VZ	2000-04-26	2021-09-08
EXC	2000-04-20	2021-09-08	WBA	2000-04-20	2021-09-08
GILD	2000-04-20	2021-09-08	WFC	2000-04-20	2021-09-08

B. *Class Labelling*

Predicting the stock market price trend is a classification problem. Therefore, we needed to generate the class labels to present our problem. Consequently, we considered three classes: Uptrend, Downtrend and Hold. The trader module trades buy, sell, and hold based on these classes. Algorithm 1 shows the labelling method [13]. In summary, within every 11 days window, if the maximum price or minimum price spots at the center of the window, it is considered a sell or buy class, respectively. The window size is a hyperparameter with 11 days as the best window size for short-term investment.

Algorithm 1 Labeling Method

```
procedure Labeling()
window Size = 11 days
while(counter Row < number Of Days In File)
        counter Row ++
If (counter Row > window Size)
    Window Start Index = counter Row – window Size
    Window End Index = window Start Index + window Size − 1
    Window Middle Index = (window Start Index + window End Index)/2
    for (i = window Begin Index;i <= window End Index;i ++)
        number = closePriceList. get(i)
        if (number < min)
          min = number
          minIndex = close Price List. Index Of min
        if (number > max)
          max = number
          max Index = close Price List. indexOf max
    if (max Index == window Middle Index)
          result="SELL"
    elif (min Index == window Middle Index)
      result="BUY"
    else
      result="HOLD"
```

C. *Technical Indicators*

Technical indicators (TI) are statistical tools that represent the stock market status from different aspects like market momentum, massive changes and many other types of information that are essential for all investors and traders. There are many technical indicators; in this paper, we used some of the well-known and later used these TIs for generating RGB images. We deployed the TIs that are summarized in Table 3.

Table 3 List of TIs used to generate images

CATEGORY	TECHNICAL INDICATORS	INTERVAL
VOLUME TECHNICAL ANALYSIS	Accumulation Distribution Index	NA
	Balance Volume	NA
	Chaikin Money Flow	NA
	Force Index	6 - 21 days
	Money Flow Indicator	6 - 21 days
	Ease of Movement	6 - 21 days
	Volume Price Trend	NA
	Negative Volume Index	NA
	Volume Weighted Average Price	6 - 21 days
VOLATILITY TECHNICAL ANALYSIS	Average True Range	6 - 21 days
	Bollinger Bands	6 - 21 days
	Keltner Channel	6 - 21 days
	Donchian Channel	6 - 21 days
	Ulcer Index	6 - 21 days
TREND TECHNICAL ANALYSIS	MACD	6 - 21 days
	SMA	6 - 21 days
	EMA	6 - 21 days
	Average Directional Movement Index (ADX)	6 - 21 days
	Vortex Indicator	6 - 21 days
	TRIX Indicator	6 - 21 days
	Mass Index	6 - 21 days
	CCI Indicator	6 - 21 days
	DPO Indicator	6 - 21 days
	KST Indicator	NA
	Ichimoku Indicator	NA
	Aroon Indicator	6 - 21 days
	PSAR Indicator	NA

	Schaff Trend Cycle (STC)	NA
MOMENTUM TECHNICAL ANALYSIS	Relative Strength Index (RSI)	6 - 21 days
	Stoch RSI (StochRSI)	6 - 21 days
	TSI Indicator	6 - 21 days
	Ultimate Oscillator	NA
	Stoch Indicator	6 - 21 days
	Williams R Indicator	NA
	Awesome Oscillator	6 - 21 days
	KAMA	NA
	Rate Of Change	6 - 21 days
	Percentage Price Oscillator	6 - 21 days
	Percentage Volume Oscillator	6 - 21 days
OTHERS TECHNICAL ANALYSIS	Daily Return	NA
	Daily Log Return	NA
	Cumulative Return	NA

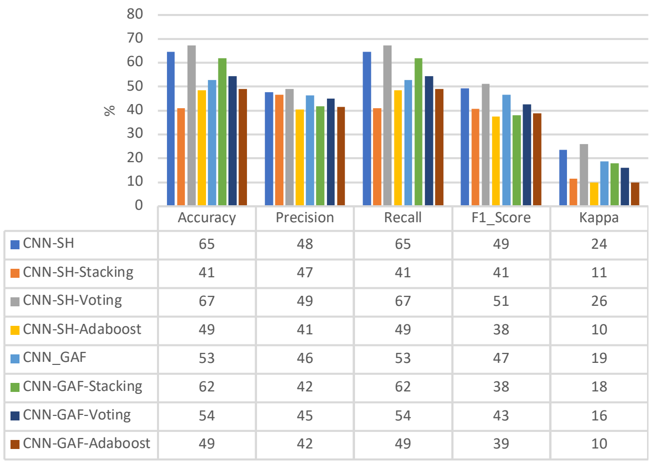
Some of the TIs get intervals to generate the output. Therefore, we considered the smallest window as five days and the largest as 21 days to generate a series of information for data formating. However, some TIs do not accept any window size, so we kept their default setting. Totally 891 TIs are generated.

D. *Results and Discussion*

We deployed two CNN-based learners with two time-series to image methods: SH and GAF. We built three ensemble models for each CNN leaner: Stacking, Voting and Adaboost. Eight models are developed, and the following charts show the results. Figure 10 presents the computational evaluation of models. We see a pattern in all five computational metrics. The CNN-SH-Voting and CNN-GAF-Stacking models' accuracy, recall, and Kappa are almost higher than others, and interestingly in Figure 11, they have the most financial return. On the other hand, the CNN-SH-Adaboost and CNN-GAF-Adaboost models have the lowest computational value and are almost the least profitable model. This pattern is also applicable for Stackings and CNN-SH, but not CNN-GAF. Therefore, we can infer that computational performance is positively correlated with financial performance.

We can compare the financial performance of individual models with each other and with ensemble models. CNN-SH group has better performance than CNN-SH group. Therefore, the GAF method brought a significant improvement to the financial return. Nevertheless, in some cases, the computational results are lower than SH. As Figure 11 presents, CNN-SH-Voting has better performance in the CNN-SH group, and CNN-GAF-Stacking has the best performance among both groups. CNN-SH has better financial performance than CNN-SH-Stacking and CNN-SH-Adaboost. Moreover, CNN-GAF ensemble models perform better than the corresponding CNN-SH ensemble models except voting.

The robustness of the ensemble models toward the stock market crash, especially during the COVID-19 pandemic, is the most interesting result in this paper. For instance, Figure 14 shows the trading chart of AFL and AIV stock between June 2019 to August 2021. The blue line is the account balance in USD, the red line is the BaH strategy representing market price, and the small green dots are the transaction fees. It can be shown that during the starting of the COVID-19 pandemic (Feb 2020 – May 2020), the model prevented losses and generated significant gains during the crash by taking several short positions. This is a critical factor of a model to be profitable and reduce investment risk during the market crash. It is an outstanding result that was not studied before.



	Accuracy	Precision	Recall	F1_Score	Kappa
🟦 CNN-SH	65	48	65	49	24
🟧 CNN-SH-Stacking	41	47	41	41	11
⬜ CNN-SH-Voting	67	49	67	51	26
🟨 CNN-SH-Adaboost	49	41	49	38	10
🟦 CNN_GAF	53	46	53	47	19
🟩 CNN-GAF-Stacking	62	42	62	38	18
🟦 CNN-GAF-Voting	54	45	54	43	16
🟫 CNN-GAF-Adaboost	49	42	49	39	10

Figure 10 Computational evaluation results

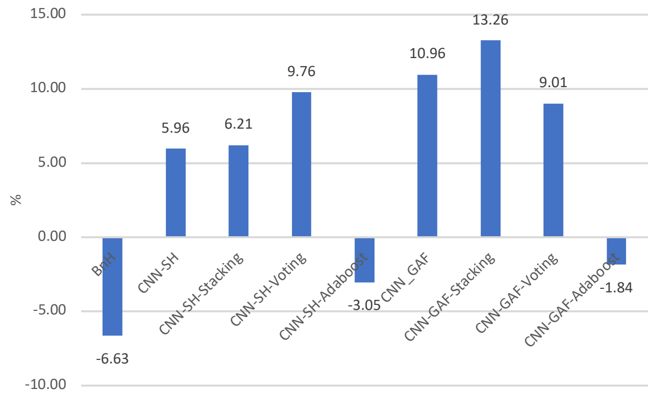


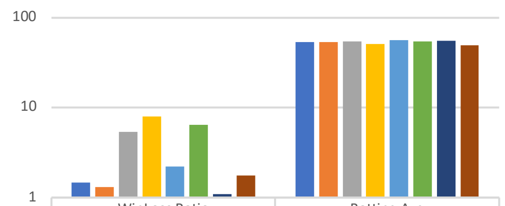
Figure 11 Average of annual return of each model

Figure 12 suggests that the Win-Loss ratio of the Voting model can be the reason for being robust during the market crash. Considering CNN-GAF financial performance in Figure 13, it can be inferred that the model has an acceptable performance during the market crash and has an outstanding performance over the up-trending period. Figure 14 shows the AFL trade result. By comparing the four charts, we can infer that, in general, ensemble models are more robust to market fluctuations and are more reliable when the market crash. Therefore it is a good practice to use ensemble models when lowering the investment risk is important.

Considering the average return per transaction, despite the CNN-SH-Stacking model having the highest return per transaction, it detects very few tractions that its financial return is lower than CNN-SH overall. Moreover, CNN-GAF has twice return per transaction than CNN-SH, which annually has 5% more returns than CNN. Last but not least, it might be inferred that CNN-GAF is the best model to use based on its performance. However, despite having good financial performance, computationally is very expensive.

VI. Conclusions and Future Directions

The results obtained from modelling by CNN show that the use of hybrid models greatly increases the resistance of the model in critical stock market conditions and can be very positive for investors in these critical conditions. Also, by using the GAF technique in preprocessing stage, the efficiency of the models increased. However, it should be noted that the computational cost of the CNN-GAF methods is much higher than the CNN-SH methods due to the input image dimensionality.



	WinLoss Ratio	Batting Avg
🟦 CNN-SH	1.47	53.6
🟧 CNN-SH-Stacking	1.3	53.4
⬜ CNN-SH-Voting	5.38	54.5
🟨 CNN-SH-Adaboost	7.96	50.8
🟦 CNN_GAF	2.23	56
🟩 CNN-GAF-Stacking	6.5	54.3
🟦 CNN-GAF-Voting	1.09	55.2
🟫 CNN-GAF-Adaboost	1.77	49.3

Figure 12 Win-Loss Ratio and Batting Average

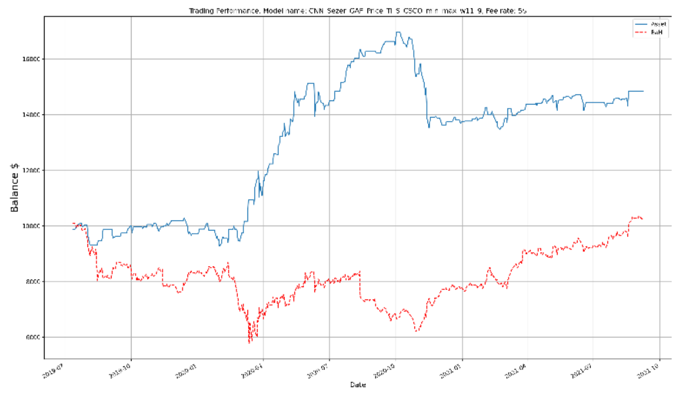


Figure 13 CNN-GAF financial return for Cisco

Future directions include 1) More investigation on the combinations and order of TIs to generate more distinguishable images in each class. 2) Since CCN-Adaboost is a very powerful learner, it can be redesigned to have more capability to learn and detect opportunities. 3) Using CNN-GAF-Stacking as the best classifier in this paper with other deep learning methods like LSTM to take advantage of the time series nature of images and enhance the model performance.

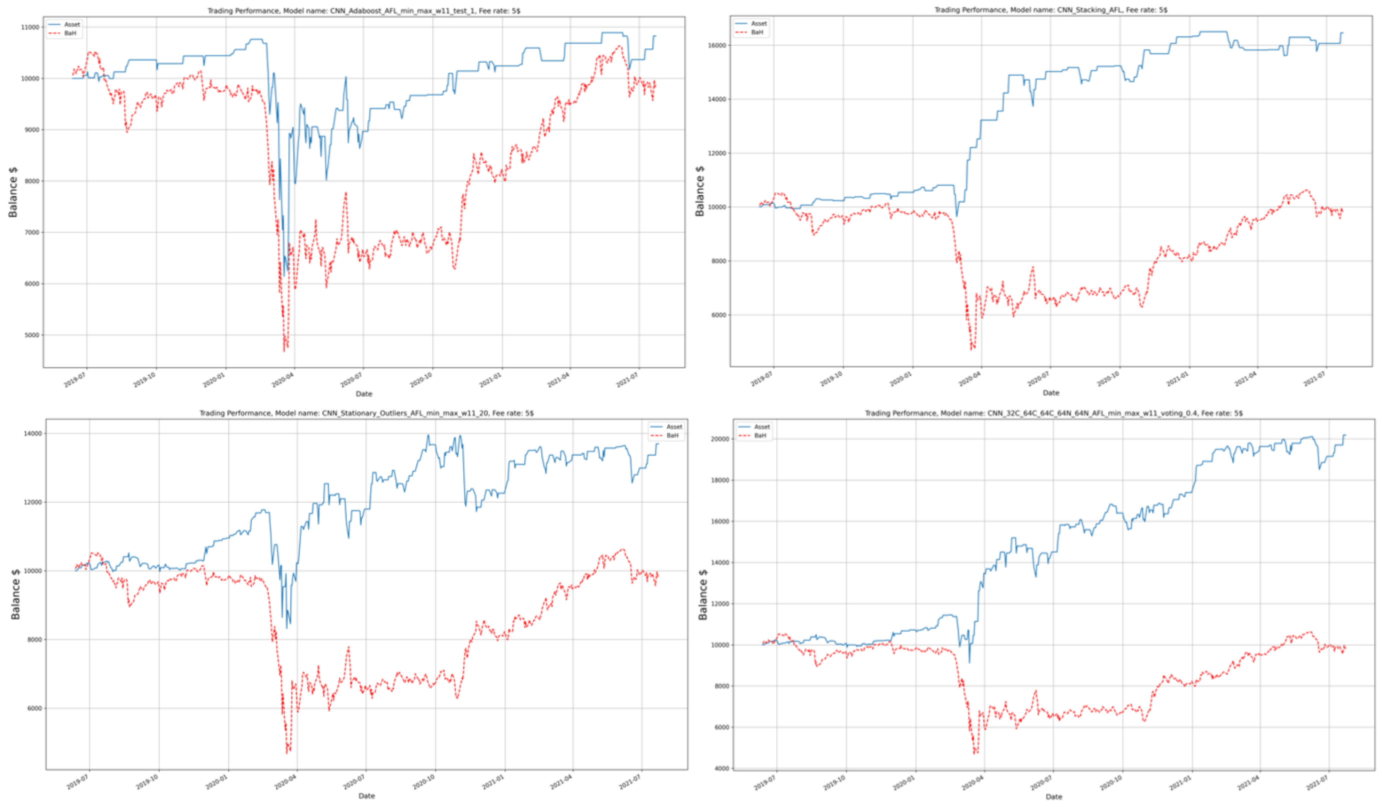


Figure 14 AFL trading results: a) Single CNN, b) Stacking-CNN, c) Adaboost-CNN, d) Voting-CNN

References

[1] M. Nabipour, P. Nayyeri, H. Jabani, A. Mosavi, E. Salwana, and S. Shahab, "Deep learning for stock market prediction," *Entropy*, vol. 22, no. 8, p. 840, Jul. 2020, doi: 10.3390/E22080840.

[2] W. Lu, J. Li, J. Wang, and L. Qin, "A CNN-BiLSTM-AM method for stock price prediction," *Neural Comput. Appl.*, vol. 33, no. 10, pp. 4741–4753, May 2020, doi: 10.1007/s00521-020-05532-z.

[3] W. Lu, J. Li, Y. Li, A. Sun, and J. Wang, "A CNN-LSTM-based model to forecast stock prices," *Complexity*, vol. 2020, pp. 1–10, Nov. 2020, doi: 10.1155/2020/6622927.

[4] S. Barra, S. M. Carta, A. Corriga, A. S. Podda, and D. R. Recupero, "Deep learning and time series-To-image encoding for financial forecasting," *IEEE/CAA J. Autom. Sin.*, vol. 7, no. 3, pp. 683–692, May 2020, doi: 10.1109/JAS.2020.1003132.

[5] I. E. Livieris, E. Pintelas, S. Stavroyiannis, and P. Pintelas, "Ensemble Deep learning models for forecasting cryptocurrency time-series," *Algorithms*, vol. 13, no. 5, p. 121, May 2020, doi: 10.3390/A13050121.

[6] M. Shi and Q. Zhao, "Stock market trend prediction and investment strategy by deep neural networks," in *2020 11th International Conference on Awareness Science and Technology, iCAST 2020*, Dec. 2020, pp. 1–6, doi: 10.1109/iCAST51195.2020.9319488.

[7] F. Zhou, H. min Zhou, Z. Yang, and L. Yang, "EMD2FNN: A strategy combining empirical mode decomposition and factorization machine based neural network for stock market trend prediction," *Expert Syst. Appl.*, vol. 115, pp. 136–151, Jan. 2019, doi: 10.1016/j.eswa.2018.07.065.

[8] W. Tovar, "Deep Learning Based on Generative Adversarial and Convolutional Neural Networks for Financial Time Series Predictions," Aug. 2020, Accessed: Oct. 18, 2021. [Online]. Available: http://arxiv.org/abs/2008.08041.

[9] E. Hoseinzade and S. Haratizadeh, "CNNpred: CNN-based stock market prediction using a diverse set of variables," *Expert Syst. Appl.*, vol. 129, pp. 273–285, Sep. 2019, doi: 10.1016/j.eswa.2019.03.029.

[10] O. B. Sezer, M. Ozbayoglu, and E. Dogdu, "A Deep Neural-Network Based Stock Trading System Based on Evolutionary Optimized Technical Analysis Parameters," *Procedia Comput. Sci.*, vol. 114, no. 2016, pp. 473–480, 2017, doi: 10.1016/j.procs.2017.09.031.

[11] F. De Arriba-Perez, S. Garcia-Mendez, J. A. Regueiro-Janeiro, and F. J. Gonzalez-Castano, "Detection of Financial Opportunities in Micro-Blogging Data with a Stacked Classification System," *IEEE Access*, vol. 8, pp. 215679–215690, 2020, doi: 10.1109/ACCESS.2020.3041084.

[12] S. Carta, A. Corriga, A. Ferreira, A. S. Podda, and D. R. Recupero, "A multi-layer and multi-ensemble stock trader using deep learning and deep reinforcement learning," *Appl. Intell.*, vol. 51, no. 2, pp. 889–905, Feb. 2021, doi: 10.1007/s10489-020-01839-5.

[13] O. B. Sezer and A. M. Ozbayoglu, "Algorithmic financial trading with deep convolutional neural networks: Time series to image conversion approach," *Appl. Soft Comput. J.*, vol. 70, pp. 525–538, Sep. 2018, doi: 10.1016/j.asoc.2018.04.024.

[14] S. Tabar, S. Sharma, and D. Volkman, "A new method for predicting stock market crashes using classification and artificial neural networks," 2020. doi: 10.1504/ijbda.2020.108697.

[15] X. Zhong and D. Enke, "A comprehensive cluster and classification mining procedure for daily stock market return forecasting," *Neurocomputing*, vol. 267, pp. 152–168, Dec. 2017, doi: 10.1016/j.neucom.2017.06.010.

[16] M. Wen, P. Li, L. Zhang, and Y. Chen, "Stock market trend prediction using high-order information of time series," *IEEE Access*, vol. 7, pp. 28299–28308, 2019, doi: 10.1109/ACCESS.2019.2901842.

[17] X. Yan and J. Zhao, "Application of improved convolution neural network in financial forecasting," in *2019 IEEE 4th International Conference on Cloud Computing and Big Data Analytics, ICCCBDA 2019*, Apr. 2019, pp. 321–326, doi: 10.1109/ICCCBDA.2019.8725661.

[18] H. Gunduz, Y. Yaslan, and Z. Cataltepe, "Intraday prediction of Borsa Istanbul using convolutional neural networks and feature correlations," *Knowledge-Based Syst.*, vol. 137, pp. 138–148, Dec. 2017, doi: 10.1016/j.knosys.2017.09.023.

[19] T. Kim and H. Y. Kim, "Forecasting stock prices with a feature fusion LSTM-CNN model using different representations of the same data," *PLoS One*, vol. 14, no. 2, p. e0212320, Feb. 2019, doi: 10.1371/journal.pone.0212320.

[20] X. Shi, Z. Chen, H. Wang, D. Y. Yeung, W. K. Wong, and W. C. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," in *Advances in Neural Information Processing Systems*, 2015, vol. 2015-Janua, pp. 802–810.

[21] J. Eapen, A. Verma, and D. Bein, "Improved big data stock index prediction using deep learning with CNN and GRU," *Int. J. Big Data Intell.*, vol. 7, no. 4, p. 202, 2020, doi: 10.1504/ijbdi.2020.113868.

[22] J. M. T. Wu, Z. Li, N. Herencsar, B. Vo, and J. C. W. Lin, "A graph-based CNN-LSTM stock price prediction algorithm with leading indicators," *Multimed. Syst.*, vol. 1, p. 3, Feb. 2021, doi: 10.1007/s00530-021-00758-w.

[23] A. Taherkhani, G. Cosma, and T. M. McGinnity, "AdaBoost-CNN: An adaptive boosting algorithm for convolutional neural networks to classify multi-class imbalanced datasets using transfer learning," *Neurocomputing*, vol. 404, pp. 351–366, 2020, doi: 10.1016/j.neucom.2020.03.064.

[24] Z. Wang and T. Oates, "Encoding time series as images for visual inspection and classification using tiled convolutional neural networks," in *AAAI Workshop - Technical Report*, 2015, vol. WS-15-14, pp. 40–46, Accessed: May 31, 2021. [Online]. Available: www.aaai.org.

[25] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, Massachusetts: The MIT Press, 2016.

[26] Wikipedia, "Convolutional neural network - Wikipedia," 2020. https://en.wikipedia.org/wiki/Convolutional_neural_network (accessed Oct. 25, 2021).

[27] W. Kenton, "Win/Loss Ratio Definition," *Investopedia*, 2019. https://www.investopedia.com/terms/w/win-loss-ratio.asp (accessed Feb. 16, 2022).

[28] W. KENTON, "Batting Average Definition," *Investopedia*, 2020. https://www.investopedia.com/terms/b/batting-average.asp (accessed Oct. 28, 2021).

[29] JAMES CHEN, "Annual Return Definition, Investopedia," *Investopedia*, 2020. http://www.investopedia.com/terms/a/annual-return.asp (accessed Feb. 16, 2022).

[30] A. Ghasemieh and R. Kashef, "Deep Learning Vs. Machine Learning in Predicting the Future Trend of Stock Market Prices," 2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC), 2021, pp. 3429-3435, doi: 10.1109/SMC52423.2021.9658938.

[31] Aasi, B., Imtiaz, S. A., Qadeer, H. A., Singarajah, M., & Kashef, R. (2021, April). Stock Price Prediction Using a Multivariate Multistep LSTM: A Sentiment and Public Engagement Analysis Model. In 2021 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS) (pp. 1-8). IEEE.

[32] Close, L., & Kashef, R. (2020). Combining artificial immune system and clustering analysis: A stock market anomaly detection model. Journal of Intelligent Learning Systems and Applications, 12(04), 83.