A deep learning based stock trading model with 2-D CNN trend detection

Conference Paper · November 2017

DOI: 10.1109/SSCI.2017.8285188

CITATIONS
19

READS
8,257

3 authors:

Ugur Gudelek
TOBB University of Economics and Technology
12 PUBLICATIONS 37 CITATIONS
SEE PROFILE

Arda Boluk
Middle East Technical University
6 PUBLICATIONS 24 CITATIONS
SEE PROFILE

Murat Ozbayoglu
TOBB University of Economics and Technology
99 PUBLICATIONS 862 CITATIONS
SEE PROFILE

A Deep Learning based Stock Trading Model with 2-D CNN Trend Detection

M. Ugur Gudelek
TOBB University of
Economics and Technology
Department of Computer Engineering
Ankara, Turkey
Email: mgudelek@etu.edu.tr

S. Arda Boluk
TOBB University of
Economics and Technology
Department of Computer Engineering
Ankara, Turkey
Email: a.boluk@etu.edu.tr

Murat Ozbayoglu
TOBB University of
Economics and Technology
Department of Computer Engineering
Ankara, Turkey
Email: mozbayoglu@etu.edu.tr

***Abstract*—The success of convolutional neural networks in the field of computer vision has attracted the attention of many researchers from other fields. One of the research areas in which neural networks is actively used is financial forecasting. In this paper, we propose a novel method for predicting stock price movements using CNN. To avoid the high volatility of the market and to maximize the profit, ETFs are used as primary financial assets. We extract commonly used trend indicators and momentum indicators from financial time series data and use these as our features. Adopting a sliding window approach, we generate our images by taking snapshots that are bounded by the window over a daily period. We then perform daily predictions, namely, regression for predicting the ETF prices and classification for predicting the movement of the prices on the next day, which can be modified to estimate weekly or monthly trends. To increase the number of images, we use numerous ETFs. Finally, we evaluate our method by performing paper trading and calculating the final capital. We also compare our method's performance to commonly used classical trading strategies. Our results indicate that we can predict the next day's prices with 72% accuracy and end up with 5:1 of our initial capital, taking realistic values of transaction costs into account.**

I. Introduction

Financial markets are massive dynamical domains that are pretty difficult to model and predict. The complicated nature of these markets led people to come up with mathematical and statistical models that make the underlying mechanics more obvious, thus, making trading more effective. In recent decades, computational intelligence models that can learn non-linear relationships between input and output values have shown promise and researchers have been trying to incorporate these models in financial forecasting [1], [2].

Financial market data is structured as time-series data and there are works in the literature that use modern machine learning techniques as well as classical statistical models such as regression models, ARIMA, etc. to perform time-series analysis [1], [3], [4]. Convolutional neural network, on the other hand, is a relatively new architecture and there are not so many papers in the literature that use this model for financial forecasting. The ones that we know of which use this model and how they use it are briefly mentioned in Section II. Major advantage of convolutional neural networks over other machine learning approaches is that, more complex non-linear relationships can be modeled by increasing the number of network layers. Conversely, increasing the number of layers in plain feed forward neural networks causes them to over-fit the given data as the number of parameters increases pretty quickly. CNNs address this issue by convolution, pooling and dropout operations, allowing much deeper architectures.

M. Ugur Gudelek and S. Arda Boluk contributed equally to this work.

In this paper, we trained a convolutional neural network on the historical financial data. While CNNs can learn non-linear features during training, providing additional information that injects domain knowledge into the data has shown to be pretty effective [5]. Thus, instead of using only the price values, we extracted some of the most commonly used fundamental analysis indicators and included them to our feature set. These indicators help the model to grasp the underlying dynamics of the market more easily. An important requirement of CNNs is that they require much more data compared to other types of models. Including the fundamental indicators also helps us to address this problem as shown in Section 3. We chose exchange-traded fund (ETF) as the primary financial asset to avoid the high-volatility of the market and to maximize our profit. To further increase the size of our dataset and the variance of the information it contains, we used multiple ETFs as well. We evaluated the performance of our method by calculating the commonly used classical performance metrics. Additionally, a virtual paper trading system is simulated where we buy and sell stocks to demonstrate that our method significantly outperforms Buy and Hold and yields considerable amount of profit in a short term.

The remainder of this paper is organized as follows. Section II reviews existing work in the literature that use CNNs and related models for financial forecasting. Section III describes the proposed method in detail. Section IV provides description for the experimental setting and gives evaluation results and comments. Section V concludes the paper and points the reader to future research directions.

II. Related Work

Computational intelligence techniques have been used in financial applications for some time [6]. Usage of fundamental and technical analysis splits these techniques that intend to forecast the trend or future price of stock market. The latter analysis relies on historical data i.e. open, close prices, volatility of stock market so the techniques which use this analysis do.

The approaches that use technical analysis have been split into several sub categories: classical machine learning [7], recently popularized deep learning [8], evolutionary algorithms [9] or fuzzy-logic based systems [6].

Although, SVM is an optimal classifier if dataset is linearly separable assumed that data is passed through kernel trick, tuning of hyper-parameter is hard to optimize. Therefore, Yeh et al. [10] and Choudhry et al. [11] investigated optimization of SVM. While a two-stage multiple kernel learning algorithm is implemented in [10], an evolutionary algorithm is implemented in [11] to optimize hyper-parameters. However, this is not a solution for linearly inseparable problems.

Some machine learning methods are combined together to boost the performance of themselves. In [12], variation of ANN ,which has tuned hyper-parameter optimized with evolutionary algorithm, is studied. Connections between neurons are randomly selected and multiple hidden layers are employed in [12].They predicted the future price value for Citigroup and Motors Liquidation Company from S&P 500. In addition to ensemble methods, ANN with decision trees is studied in another study [13].

Persio et al. [8] has examined differences between Convolutional Neural Network, Multi-layer Perceptron(MLP) and Long Short-Term Memory(LSTM). They used historical data of S&P500 index which is available from 1950 to 2016. They created 30 day of sample chunks by sliding window over time series to predict the value of 31st day. Although, the power of technical analysis is not used, results for CNN model are noteworthy.

Performances of many researches are calculated with mean squared error(MSE) or accuracy metrics. However, for stock market forecasting, returns of predictions are also used commonly. In this manner, we used both set of metrics to evaluate our proposed model performance.

III. Methodology

This section first introduces how we obtain dataset, then how we extract features, why we use clustering, how we construct images, and finally how we construct convolutional neural network, accordingly.

A. Dataset and Feature Extraction

To forecast the price or trend for following days, historical price values should be used essentially. To retrieve these historical data, Google finance is used. Google finance provides daily stock prices with several other features and this is the full signature of given data: date, open, high, low, close, adjusted close. In addition, they provide these features error-free i.e. adjusted in case of stock splits, dividends or rights offerings.

Financial data is highly volatile due to the nature of the stock market. To reduce this volatility, exchange-traded funds (ETF) are used. ETFs are compositions of commodity, bonds and index funds. Combining different type of indexes, bonds and commodities reveals the advantage of lower volatility while keeping the advantages of stock exchange. ETFs are bought and sold similar to regular stocks with higher liquidity and lower fees. Table I tabulates the most commonly traded ETFs in the New York Stock Exchange. Hence, these stated ETFs are used in this research.From now on throughout the paper, previously stated data will be named as raw data.

There are two types of methods used in finance to analyze financial data: fundamental analysis and technical analysis. On one hand, fundamental analysis is interested in statements of companies and other individuals and other general concepts which is related to the stock itself. This analysis requires domain expertise and interpretation of financial reports. Extracting features from fundamental analysis might require different machine learning approaches e.g. natural language processing. On the other hand, technical analysis is interested in features of stock e.g. close and volume, without using domain expertise. Latter analysis can be interpreted as time series analysis hence approaches that use time series analysis can be used e.g. recurrent neural network (RNN), long-short term memory (LSTM), convolutional neural network (CNN). In this research, we have used convolutional neural network.

To enrich the dataset, several technical analysis methods are performed. We have used relative strength index (RSI), simple moving average (SMA), the moving average convergence/divergence oscillator (MACD), the Williams percent range (Williams %R), the stochastic oscillator, the ultimate oscillator, the money flow index (MFI). Descriptions and formulas for each technical analysis methods used in research, are stated in Table II.

Generally, price values are consistently increasing over long period of time. More specifically, mean of price values increase over time. This is the reason why buy and hold strategy is the preferred model when compared to other sophisticated methods. However, statistical properties such as mean and variance are not meaningful in the long run. To be able to use mean and variances, these properties should be made constant over time i.e. stationarized. In order to make price time series stationary, first difference method is used. Simply, price values of successive days are subtracted. At the end, as a result of first difference, mean around zero-point is obtained. Figure 1 shows non-stationary and stationary price values over time. As a result of normalization with hyperbolic tangent function ($\tanh$) (3b) , stationary price values are bounded between -1 and 1.

In order to achieve class values for up and down trend, normalized and stationarized version of price values are categorized taking 0 as decision point. However, if up,steady and down are taken as classes, decision point will be unclear. To solve this issue, we split normalized and stationarized price values into 3 clusters with the intend of achieving the minimum sum of variances within clusters.

TABLE I
SELECTED ETFS AND THEIR DESCRIPTIONS

Name	Description	Inception Date	Volume
XLF	Financial Select Sector SPDR ETF	12/16/1998	71,886,065
XLU	Utilities Select Sector SPDR ETF	12/16/1998	11,342,530
XLI	Industrial Select Sector SPDR ETF	12/16/1998	8,988,538
SPY	SPDR S&P 500 ETF	1/22/1993	68,675,793
XLP	Consumer Staples Select Sector SPDR ETF	12/16/1998	9,721,714
EWG	iShares MSCI Germany ETF	3/12/1996	2,966,371
XLB	Materials Select Sector SPDR ETF	12/16/1998	4,157,434
XLK	Technology Select Sector SPDR ETF	12/16/1998	11,439,758
XLV	Health Care Select Sector SPDR ETF	12/16/1998	6,891,614
EWH	iShares MSCI Hong Kong ETF	3/12/1996	2,586,985
EWC	iShares MSCI Canada ETF	3/12/1996	2,680,153
XLY	Consumer Discret Sel Sect SPDR ETF	12/16/1998	4,257,841
EWW	iShares MSCI Mexico Capped ETF	3/12/1996	2,363,319
DIA	SPDR Dow Jones Industrial Average ETF	1/13/1998	2,510,130
XLE	Energy Select Sector SPDR ETF	12/16/1998	16,494,257
EWA	iShares MSCI Australia ETF	3/12/1996	2,151,390
EWJ	iShares MSCI Japan ETF	3/12/1996	5,998,415

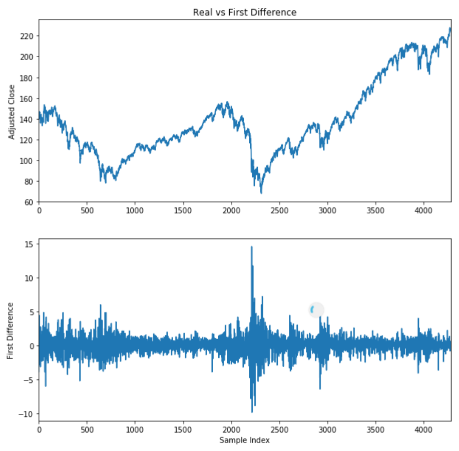


Fig. 1. Real vs First Difference

B. CNN

Convolutional neural network (CNN) is part of the family of neural network (NN) which is a variation of multilayer perceptron (MLP). CNN consists of an input layer, several hidden layers and an output layer like any other NNs. Input layer is a representation of identity function, $f(x) = x$. Output layer which makes decisions, passes previously calculated weights through a linear function (4). Hidden layers are either convolutional, pooling, dropout or fully connected. In addition, all layers have activation functions at the end which gives additional functionality e.g. normalization. $sigmoid$, $\tanh$ and $RELU$ (figures 3a, 3b, 3c respectively) are examples of these activation functions. Weights of convolution layers can be seen as 2D-filters and they apply convolution operation with these filters. Convolution operation is a process which sums the point-wise multiplications of given two functions while sliding the operation window. Pooling layer generalizes the elements in window frame while sliding this window. For example, max pooling outputs the maximum elements for a given window while sliding it. Dropout [14], selects several neurons, that feed the input of next layers and reduces over-fitting. Finally, fully connected layers can be thought as fully connected version of classical MLP. With the explanation of fully connected layers, CNNs can be seen as combination of MLP and filters which can operate as either convolution or pooling. In order to optimize weights of CNN, we have used an adaptive learning rate method (ADADELTA) optimizer [15]. In Fig 2 CNN network used in this research, is provided.
Apart from the fact that CNNs give noteworthy performance [16], they require much more data compared to other types of models. With the purpose of solving this issue, we merge all ETFs and create a satisfactory dataset for a finance related problem. Additionally, merging reveals that we do not have to stick to one set of stock only. Model trained with this dataset, assumed that it performs well, will be universal for all kinds of stocks rather than a particular stock. Result of the trained model will be examined in section IV.

$$\tanh(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}} \tag{1}$$

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2}$$

$$R(x) = max(0, x) \tag{3}$$

$$y_i = \beta_0 1 + \beta_1 x_{i1} + ... + \beta_p x_{ip} = x_i^T \beta, \qquad i = 1, ..., n \tag{4}$$

where where T denotes the transpose and $x_i^T \beta$ is the inner product between two vector.

C. Clustering

2D convolution process requires locality over both x and y axes. Despite, time series have locality on x-axis naturally, however not on y-axis. To solve this problem, we examine clustering while taking each feature time series as input. When different features are clustered together, each cluster satisfies locality within its boundary. Particularly, agglomerative clustering is selected because of visualization of dendrogram and bottom-up approach. Agglomerative clustering is a version of hierarchical clustering which solves the clustering problem iteratively and outputs the result as a dendrogram. This dendrogram can be cut at any intended level. Euclidean distance metric (5) is used while constructing distance matrix with Pearson correlation coefficient (6) and the corresponding dendrogram of features are created.

TABLE II
TECHNICAL ANALYSIS METHODS AND DESCRIPTIONS, FORMULAS

Feature Name	Description	Formula
RSI	It provides strength or weakness of stock market for recent period	$RSI = 100 - \frac{100}{1+\frac{averagegain}{averageloss}}$
SMA	It is an average of closing price for any given period. Averaging window slides over time.	$SMA(M,n) = \sum_{k=a+1}^{a+n} \frac{M(k)}{n}$
EMA	Similar to SMA however it gives more weight to latest data.	$EMA(M,t,\tau) : (M(t) - EMA(M,t-1,\tau)).\frac{2}{\tau+1} + EMA(M,t-1,\tau)$
MACD Line	It is a momentum indicator that shows relationship between series	$MACDLine(t) : (EMA(M,t,\tau_1) - EMA(M,t,\tau_2))$
MACD Histogram	It looks relationship between MACD Line and another trigger signal and provides buy-sell signal.	$MACDHistogram(t) : MACDLine(t) - EMA(MACDLine,t,\tau_3)$
William %R	It measures overbought and oversold line of market.	$R = \frac{max(high)-close}{max(high)-min(low)} * -100$
Stochastic Oscillator	Comparable with William %R to detect extreme lines	$K = \frac{close(t) - min(low(t))}{max(high(t)) - min(low(t))} * 100$ $D = SMA(K,3)$ $SO = K - D$
Ultimate Oscillator	It is used to reduce the volatility and false transactions.	$BP = close(t) - min(low(t), close(t-1))$ $TR = max(high(t), close(t-1)) - min(low(t), close(t-1))$ $UOS = 100 * (\frac{(\frac{\tau_3}{\tau_1} * \sum_{i-\tau_1}^{i} \frac{BP}{TR}) + (\frac{\tau_2}{\tau_1} * \sum_{i-\tau_2}^{i} \frac{BP}{TR}) + (\frac{\tau_1}{\tau_1} * \sum_{i-\tau_3}^{i} \frac{BP}{TR})}{(\tau_1+\tau_2+\tau_3)}$
MFI	It is a momentum indicator that uses volume data. It measures the inflow and outflow of the money over any period of time.	$TypicalPrice(t) = \frac{high(t)+low(t)+close(t)}{3}$ $RawMoneyFlow = TypicalPrice(t).Volume(t)$ $MFR(i) = \frac{(\sum_{i-\tau}^{i} PositiveMoneyFlow)}{(\sum_{i-\tau}^{i} NegativeMoneyFlow)}$ $MFI(i) = 100 - 100/(1+MFR(i))$

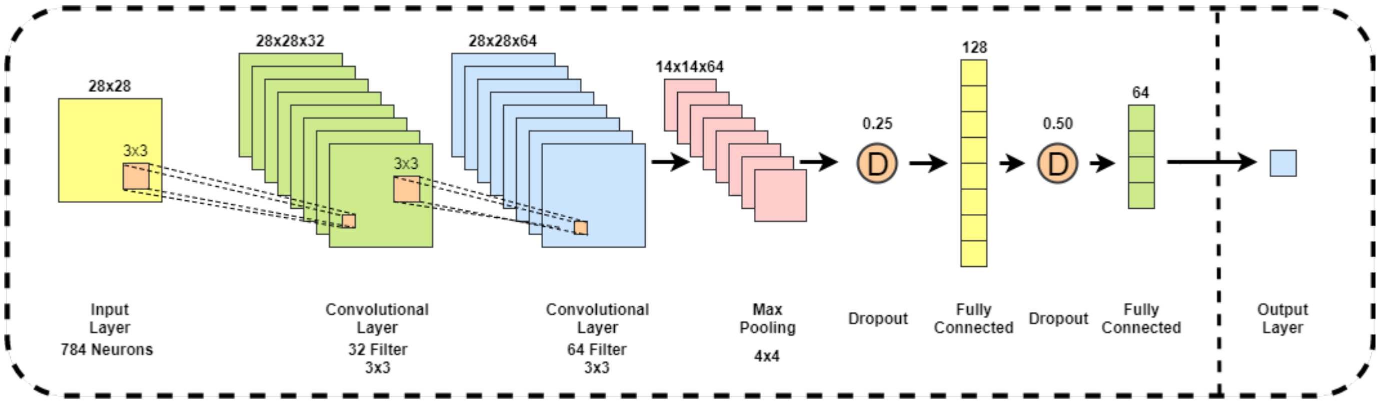


Fig. 2. Convolutional Neural Network

$$||a-b||_2 = \sqrt{\sum_{a}^{b}(a_i - b_i)^2} \quad (5)$$

$$\rho_{x,y} = \frac{cov(X,Y)}{\sigma_x \sigma_x} \quad (6)$$

where cov is the covariance, σ_x is the standard deviation of X σ_y is the standard deviation of Y

D. Images

CNN takes 2D images as input and updates weights of filters, as stated before. Hence, we need images to feed the CNN model. In order to create images, several processes are employed. Firstly, features stated in Table III are calculated

TABLE III
FEATURES

Feature Name	Parameters	Amount
tanh((stationary(close)))	-	1
volume	-	1
RSI	15-20-25-30	4
SMA	15-20-25-30	4
MACD	26,12 28,14- 30,16	3
MACD_trigger	9,26,12- 10,28,14- 11,30,16	3
William %R	14-18-22	3
Stochastic oscillator	14-18-22	3
Ultimate oscillator	7,14,28-8,16,22-9,18,36	3
MFI	14-18-22	3

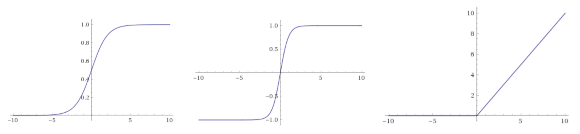

(a) sigmoid (b) tanh (c) RELU

Fig. 3. Activation functions.

for all training and test data. As a result of calculation, every day sample ended up with 28 different features. If multi-layer perceptron were to be used to predict the trend, these features would have been sufficient to develop the model. However, CNN takes a 2D input which needs to be a square matrix. Hence, we performed a sliding window approach to split this 28-line time series into 28x28 chunks. Sliding window was slid one day at a time. Each square matrix was labeled with the following day of last day used during the creation of each square matrix. Following day is selected because we want to forecast the trend of the next day of any given day. As a result, a 28-day window with different features was utilized. Therefore, patterns arise from exchanges which are performed period of week or month will be more detectable for CNN model.

TABLE IV
EXPERIMENTAL SETUP RESULTS

	Accuracy	MSE	Profit w/ commis-sion	Profit w/o commis-sion
Regression with 2 class	71.72%	0.2631	54111.13	58370.02
Regression with 3 class	49.82%	0.2631	38366.43	39780.84
Classification with 2 class	78.46%	-	38454.17	43085.58
Classification with 3 class	63.05%	-	33398.03	36100.66

IV. RESULTS

A. Experimental Setup

We evaluated our model on both classification and regression tasks. In both of these tasks, we've mapped the prices of the ETFs between -1 and 1 using tanh function before the training phase, as mentioned in Section III-A. This mapping also allowed us to interpret the predictions of the regression model as confidence values, which increases the profit at the end.

For both classification and regression tasks, the ratio of 0.9 was used as training-test set ratio. Because of the nature of the data used, we could not choose training and test samples randomly, as this would cause model to interpolate missing future prices in the training set, leading to false evaluation results. Instead, we divide the data keeping the temporal ordering of the samples intact. Regarding the training-test ratio, we determined the training period to be between 04/06/2000 and 03/23/2015 and test period 03/24/2015 and 11/17/2016, taking samples from each ETF in these periods.

In the test phase of the regression task, the target values and predictions are discretized to be able to perform trading. Two discretization schemes are adopted, namely 2-class and 3-class regression. In 2-class regression, values that are greater than 0 are considered as "buy" signals and values that are less than or equal to 0 are considered as "sell" signals. In 3-class regression, we set the values in the range [-1,-0.38] as "sell" signals, values between (-0.38, 0.38] as "hold" signals and values between (0.38, 1] as "buy" signals. The threshold values -0.38 and 0.38 split the data with minimum variance, as mentioned in Section III.

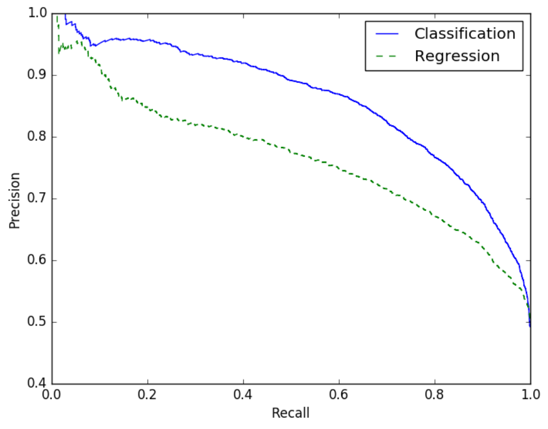
The classes in the classification task are determined in a similar manner. However, unlike regression, the determined classes are used in the training phase as well. In 2-class classification, samples with target values that are less than 0 are considered as negative samples while the rest are considered as positive. In 3-class classification, the threshold values -0.38 and 0.38 are used to determine the classes.

We evaluated our method using common classical performance metrics, namely, MSE, accuracy, Precision-Recall and Receiver Operating Characteristic (ROC) and the results are provided in the following section. We also evaluated the model using a trading algorithm where we buy and sell the ETFs at the actual prices of the test data. The algorithm is provided in Algorithm 1.

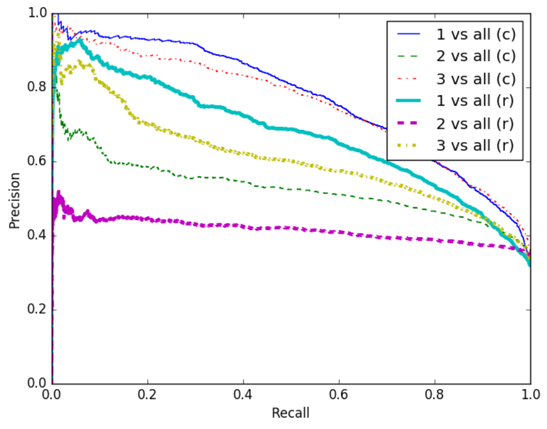
This algorithm is used for the regression models. For the classification models we can only determine the class of the samples, thus, cannot find the ETFs whose predicted prices will go highest and will go lowest.

B. Validation Results

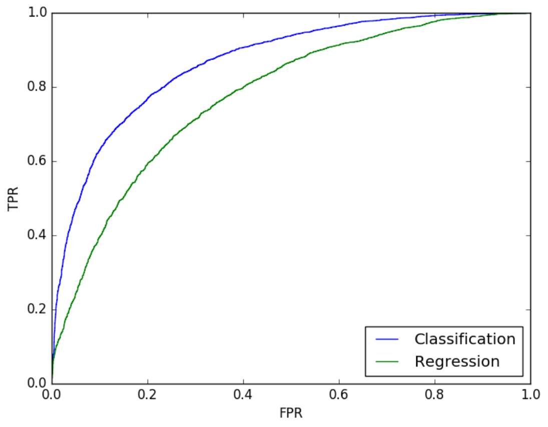
Evaluation results of the 4 experiments are described in Table IV. The MSE of the regression model shows that it can model the data with a low error. Accuracy results of 2-class regression and 2-class classification are relatively close as expected because targets of the regression model



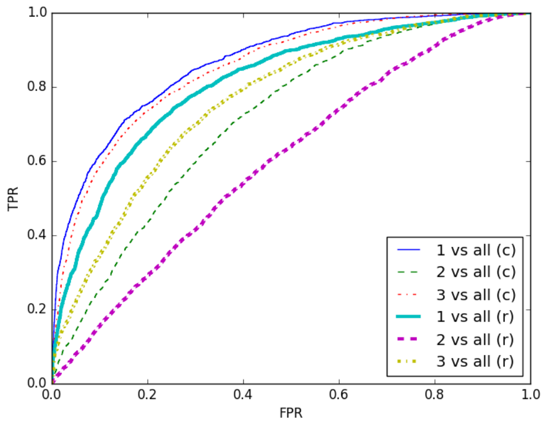
(a) PR curve for 2-class models



(b) PR curve for 3-class models



(c) ROC curve for 2-class models



(d) ROC curve for 2-class models

Fig. 4. Precision-Recall and ROC curves.

is mapped between [-1,1] similar to the probability values and evaluated as a classification model regarding the accuracy metric. The same logic goes for the 3-class regression and 3-class classification, thus, their performances are also somewhat close.

Difference between the accuracy results of 2-class and 3-class models is considerably large, both for regression and classification models. This shows that it is more difficult for the model to distinguish the cases where it should hold and the cases where it should buy or sell than the cases where it should distinguish between buying and selling. This is caused by the volatility of the prices, as the changes in the "hold" region are not stable and are randomly fluctuating to the "buy" and "sell" regions.

We provide confusion matrix in Figure 5 to show the performance of our model with default thresholds, namely, 0 for 2-class models and -0.38 and 0.38 for 3-class models. In 2-class models, classes are distinguished clearly, as the figure shows. In 3-class regression, however, class 2 is confused with classes 1 and 3 considerably. The 3-class classification model is more successful in distinguishing class 2 from other classes compared to it's regression counterpart. This shows that 3-class classification model is more suited to the 3-class task provided that it yields a confidence value, like the regression model does.

We also provide Precision and Recall curves for the evaluation of the models. Figure 4a shows that the 2-class classification and regression models have somewhat similar performances. Their precision decays with similar speeds as their True Positive Rate (TPR) is increased.

In Figure 4b, 2-vs-all curves for both classification and regression are noticeably below other curves. This shows that "hold" classes are not in balance with "buy" and "sell" classes. This also explains why the accuracy results of 3-class models are worse compared to the 2-class models.

Figure 4c shows that 2-class models perform pretty close and Figure 4d shows that the 3-class models have the worst performance regarding the "hold" classes.

Using only the classical evaluation metrics is not enough to evaluate a financial forecasting system, where there are

Algorithm 1 Confident Trading Algorithm

```
 1: procedure TRADE($Predictions, Prices, Capital$)
 2:     $min\_date \leftarrow \underset{date}{\operatorname{argmin}}\, Predictions$
 3:     $max\_date \leftarrow \underset{date}{\operatorname{argmax}}\, Predictions$
 4:     $cur\_date \leftarrow min\_date$
 5:     while $cur\_date < max\_date$ do
 6:         $higher\_stocks \leftarrow []$
 7:         $lower\_stocks \leftarrow []$
 8:         for $stock \in Predictions[cur\_date]$ do
 9:             if stock.predicted $> 0$ then
10:                 higher_stocks.add(stock)
11:             else
12:                 lower_stocks.add(stock)
13:             end if
14:         end for
15:         if len(higher_stocks) $\neq 0$ then
16:             $h\_stock \leftarrow \underset{predicted}{\operatorname{argmax}}\, higher\_stocks$
17:             if $Capital >$ Prices[h_stock][cur_date] then
18:                 Buy h_stock with all the $Capital$ left
19:                 Update $Capital$
20:             end if
21:         end if
22:         for $l\_stock \in lower\_stocks$ do
23:             Sell l_stock
24:             Update $Capital$
25:         end for
26:     end while
27:     return $Capital$
28: end procedure
```

potential practical applications. Thus, we evaluated our method using a trading system, whose algorithm is given in Algorithm 1, on the test data. Table IV shows the profits our method obtained at the end of the trading term, with and without taking the transaction cost into account which is 5 USD per transaction. Our initial capital is 10000 USD.

Profit is considerably higher for the regression models than the classification models. The reason for this is due to the fact that the confident trading algorithm for the regression models use the confidence value the model provides, while the classification models do not do this and buy all the stocks that it thinks will rise, somewhat naively.

The difference between the profits that the 2-class and 3-class regression models yield is consistent with their accuracy results and do not need additional consideration. However, the difference between the profits that the classification models yield is low considering that their accuracy results differ radically. This can be explained by the fact that the classification models do not provide confidence values, thus, the difference between the profits are averaged out when the trading algorithm treats all the ETFs, that are in the same predicted class, equally.

Finally, we compare our model to the Buy-and-Hold model

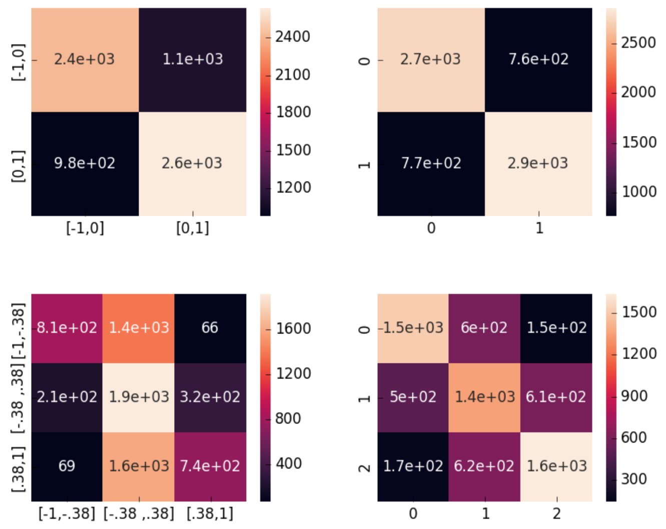


Fig. 5. Confusion Matrix for all the models. The figures on the first column shows 2 and 3-class regression results, respectively. The figures on the second column shows 2 and 3-class classification results.

and provide our results in Table V.

The columns in the table are in the following sequence from left to right, total capital with our proposed strategy, total capital with buy and hold strategy, our annualized return, buy and hold annualized return, annualized number of transactions, percent of successful transactions, average percent profit per transaction, average transaction length in days, maximum transaction profit percentage, maximum transaction loss percentage, maximum capital during the test period and minimum capital during the test period.

When the results are analyzed, the proposed model was able to outperform Buy & Hold for almost all of the chosen ETFs (with the exception of SPY). Furthermore, when we used all ETFs together in a basket of ETFs approach (represented as ALL in the table), we were able to boost the profits even further. When the percent success rate of transactions are observed, it is seen that almost all of the ETFs performed over 70%, (with an average of 72.9%) resulting in high profits. Even though this is a preliminary work, the achieved results look promising.

V. Conclusion

We developed a deep learning based Stock Trend Forecasting model in this study. The proposed model utilizes 2-D Convolutional Neural Network for trend detection. Unlike other computational intelligence based stock forecasting models, this particular model generates 2-D images from technical analysis time series values in x-axis and the clustered and sequenced feature values in y-axis. The developed model is trained using 17 different most commonly used ETFs between 2000-2014 and tested between 2015-2017. The performance evaluation results indicate that the proposed model has high precision,recall, accuracy values; and when used as part of a

TABLE V

COLUMN DESCRIPTIONS ARE PROVIDED IN THE LAST PART OF SECTION IV

NAME	OUR	BAH	OURr	BAHr	ANT	POS	APT	L	MPT	MLT	MAXc	MINc
ALL	54111.13	9853.37	100.67	-0.88	94.64	79.87	2.90	8.24	54.21	-0.04	54111.13	10119.69
spy	8549.28	10691.37	-9.30	3.98	52.38	39.77	-0.12	3.44	3.16	-0.05	10287.12	8431.75
xlf	30541.15	9540.80	66.55	-2.80	57.14	79.17	1.22	3.43	10.85	-0.05	30541.15	10065.06
xlu	30206.49	10909.44	65.89	5.18	51.19	84.88	1.33	3.95	5.91	-0.01	30206.49	10114.85
xle	46492.65	9893.89	91.62	-0.63	61.90	73.08	1.55	2.94	15.24	-0.03	46492.65	10119.69
xlp	21508.94	10608.85	45.65	3.52	60.12	83.17	0.80	3.03	4.92	-0.04	21508.94	9996.15
xli	23959.47	11021.29	52.08	5.79	61.31	72.82	0.90	2.94	8.19	-0.04	24016.39	9963.00
xlv	27594.79	9311.50	60.50	-4.25	59.52	79.00	1.06	3.40	8.36	-0.04	27644.09	9916.56
xlk	27835.00	11400.60	61.02	7.80	60.12	82.18	1.06	3.22	7.65	-0.04	27835.00	9997.23
ewj	32804.14	9515.00	70.81	-2.96	59.52	84.00	1.24	3.42	8.26	-0.04	32804.14	9871.80
xlb	34490.29	10127.89	73.80	0.76	59.52	78.00	1.30	2.99	12.84	-0.02	34490.29	10024.85
xly	26507.72	10675.96	58.10	3.89	55.95	78.72	1.09	3.31	7.92	-0.04	26507.72	9927.49
eww	43569.93	7359.73	87.74	-18.24	57.14	78.12	1.60	2.98	9.04	-0.03	43569.93	10026.12
dia	22815.33	10972.40	49.16	5.52	58.93	80.81	0.88	3.22	5.11	-0.04	22815.33	10033.12
ewg	32702.85	8716.90	70.63	-8.17	51.79	85.06	1.42	3.85	7.41	-0.02	32702.85	9903.68
ewh	35980.31	9044.16	76.32	-5.98	51.79	79.31	1.53	3.62	12.50	-0.03	35994.56	11204.94
ewc	32397.56	9404.06	70.07	-3.66	58.93	79.80	1.24	2.89	11.55	-0.01	32397.56	10333.17
ewa	43791.09	8393.68	88.04	-10.42	52.98	83.15	1.73	3.27	11.78	-0.01	43791.09	9873.36

trading model significantly outperforms Buy & Hold Strategy. Future work includes optimizing the feature selections and values, increasing the image size, buy-sell strategy algorithm optimization.

ACKNOWLEDGMENT

This study is funded by TUBITAK (The Scientific and Technological Research Council of Turkey) through project grant no. 215E248.

REFERENCES

[1] R. C. Cavalcante, R. C. Brasileiro, V. L. Souza, J. P. Nobrega, and A. L. Oliveira, "Computational intelligence and financial markets: A survey and future directions," *Expert Systems with Applications*, vol. 55, pp. 194–211, 2016.

[2] B. Krollner, B. Vanstone, and G. Finnie, "Financial time series forecasting with machine learning techniques: A survey," 2010.

[3] A. A. Adebiyi, A. O. Adewumi, and C. K. Ayo, "Comparison of arima and artificial neural networks models for stock price prediction," *Journal of Applied Mathematics*, vol. 2014, 2014.

[4] B. Kelly and S. Pruitt, "The three-pass regression filter: A new approach to forecasting using many predictors," *Journal of Econometrics*, vol. 186, no. 2, pp. 294–316, 2015.

[5] J. Heaton, "An empirical analysis of feature engineering for predictive modeling," in *SoutheastCon, 2016*. IEEE, 2016, pp. 1–6.

[6] G. S. Atsalakis and K. P. Valavanis, "Surveying stock market forecasting techniques Part II: Soft computing methods," *Expert Systems with Applications*, vol. 36, no. 3, pp. 5932–5941, apr 2009. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0957417408004417

[7] O. B. Sezer, A. M. Ozbayoglu, and E. Dogdu, "An Artificial Neural Network-based Stock Trading System Using Technical Analysis and Big Data Framework," in *Proceedings of the SouthEast Conference on - ACM SE '17*. New York, New York, USA: ACM Press, 2017, pp. 223–226. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3077286.3077294

[8] L. Di Persio and O. Honchar, "Artificial neural networks architectures for stock price prediction: Comparisons and applications," *International Journal of Circuits, Systems and Signal Processing*, 2016.

[9] U. Sahin and A. M. Ozbayoglu, "TN-RSI: Trend-normalized RSI Indicator for Stock Trading Systems with Evolutionary Computation," *Procedia Computer Science*, vol. 36, pp. 240–245, 2014. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1877050914013350

[10] C.-Y. Yeh, C.-W. Huang, and S.-J. Lee, "A multiple-kernel support vector regression approach for stock market price forecasting," *Expert Systems with Applications*, vol. 38, no. 3, pp. 2177–2186, 2011. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0957417410007876

[11] R. Choudhry and K. Garg, "A Hybrid Machine Learning System for Stock Market Forecasting," *World Academy of Science, Engineering and Technology*, vol. 2, no. 15, pp. 315–318, 2008.

[12] 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*, vol. 39, no. 1, pp. 611–620, 2012. [Online]. Available: http://dx.doi.org/10.1016/j.eswa.2011.07.051

[13] S.-P. Tsai, C.-F. and Wang, "Stock Price Forecasting by Hybrid Machine Learning Techniques," *Proceedings of the International MultiConference of Engineers and Computer Scientists*, vol. 1, 2009. [Online]. Available: http://www.iaeng.org/publication/IMECS2009/IMECS2009_pp755-760.pdf

[14] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014. [Online]. Available: http://www.jmlr.org/papers/volume15/srivastava14a.old/source/srivastava14a.pdf

[15] M. D. Zeiler, "ADADELTA: An Adaptive Learning Rate Method," dec 2012. [Online]. Available: https://arxiv.org/abs/1212.5701

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," pp. 1097–1105, 2012. [Online]. Available: https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks