DEEP LEARNING VISION APPROACH TO PREDICTING STOCK PRICE MOVEMENT OUT OF CONSOLIDATION PATTERNS

Justin Kennedy

Department of Data Science
Columbia University
Email: jak2294@columbia.edu

ABSTRACT

In this study a convolutional neural network framework is used to predict the future direction and level of volatility for a stock's price using an image of the stock's recent price chart. Historical instances of a specific technical price pattern are used for training. An online stock scanning screen is used to identify stocks within the last three years that fit the particular pattern criteria which in general terms includes a previous uptrend into a price consolidation. The vision approach proves to effectively predict future price direction and volatility on the given testing sets, producing classification accuracies significantly greater than random predictions. The algorithm in the context in which it's applied represents a potentially novel approach in the field for stock price prediction to be explored further in future work.

Index Terms— stock trading systems, deep learning, vision, convolution neural network applications

1. INTRODUCTION

From corporate traders to armchair analysts, technical analysis has become a widely employed approach to understanding stock price movements. By valuing simply the trending of stock price patterns over a period of time to develop trading strategies, technical analysis represents a company fundamental and macroeconomic agnostic approach to investing. The basis behind technical analysis relies on the fact that price changes can be attributed to changes in demand (and to a lesser extent supply). Certain price patterns have been publicized over the years as being predictive of future price movement by representing particular shifts in demand, including the 'double bottom' and 'tea cup' patterns.

One such pattern occurring in different forms frequently throughout stock price histories is a price consolidation. In simple terms, a price consolidation in this context refers to an initial high in price, followed by a set low, and subsequent oscillation between the high and low with lower and lower amplitudes (similar to damped harmonic oscillation). An example of a price chart with multiple consolidations is shown in Figure 1. Consolidations in stock prices over a set period of time have been known to be highly conducive to volatility spikes due to a 'drying out' of selling pressure in between support and resistance points (represented by the formed highs and lows). At these points, a rise in demand for the stock in terms of volume can drastically move the stock up; likewise, sharp selling on the tail end of a consolidation can dip the price significantly, especially for stocks with generally low volume such as growth stocks. The general pattern is well documented by traders such as William O'Neil and Mark Minervini.

Figure 1. The price chart for WWE from July 2018 to October 2018 is shown with two examples of price consolidations circled.

In this study I look to emulate a technical analyst's approach of scanning stock charts for patterns through use of a convolutional neural network to classify given stock charts as positive, negative, or neutral based off their expected returns. Each stock chart used in training uses a buy point off of a price consolidation pattern variant. I look to use the visual information in the form of sequential daily price movements in a time series plot (as well as the colors associated with the direction of each daily price change) to use in prediction of the stock price direction and magnitude in the weeks following the price consolidation breakout. I am particularly interested in whether a vision approach can add any value to a more standard deep learning prediction approach with the goal of converting the network probability

outputs into a robust trading system. Particularly in regards to price consolidation patterns, I look to see if certain consolidations of the presented variant are 'better' than others. Additionally, I look to use convnet filter visualizations to identify what properties of the charts the network may be picking up on as useful prediction features for the given output classes.

2. RECENT WORKS

In recent years, deep learning approaches to predicting stock price movements have become very popular across the financial industry. From NLP methods to common regression neural networks using raw time series analysis, many approaches have been tested with an aim to predict stock price direction and form accompanying trading systems that maximize the expected return (with minimum drawdown) utilizing the outputted probabilities of the developed models. However, while the use of deep learning networks for pattern recognition has increased heavily in recent years due to their superior accuracy to shallow networks in most cases, vision methods to understanding stock price movements have been relatively unexplored. The system in [1] represents a convolutional neural network approach to identifying instances of particular price patterns, namely the 'double top' and 'double bottom' patterns, through analysis of stock chart images. Using a CNN and LSTM recurrent network, they were able to effectively identify instances of these patterns. However, the problem of predicting the future return proceeding these patterns is not addressed.

3. SYSTEM DESCRIPTION

The assets used for training and testing in this study were solely US equities. The StockCharts.com stock scanning algorithm was used to identify stocks with a particular price pattern in the past three years. Specifically, the detected pattern represented a typical growth stock cycle with an uptrend of over 20% followed by a price consolidation. Growth stocks are defined here as fast growing companies, represented in their price charts as sequences of up trends followed by consolidations. Some of the scanning criteria are shown in Figure 3.

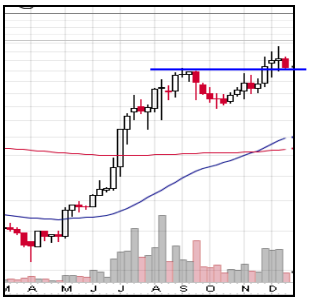


Figure 2. Example of a stock whose price pattern was detected using the StockCharts.com stock scanning. The buy point for the purpose of this study is the breakout from the consolidation high as shown with the blue line.

- 200-day Simple Moving Average of Close for 30 days ago is less than or equal to 200-day Simple Moving Average of Close for today
- Daily Close for today is greater than or equal to Daily Upr Price Channel(50) for today * 0.95
- Daily Upr Bollinger Band(10,2) for today is greater than or equal to Daily Upr Price Channel(5) for today
- Daily Lwr Bollinger Band(10,2) for today is less than or equal to Daily Lwr Price Channel(5) for today
- Daily Upr Price Channel(5) for today is less than or equal to Daily Upr Keltner Channel(10,1,10) for today

Figure 3. A subset of the criteria used in stock scanning for the desired pattern is shown. The Bollinger Band and Keltner channel criterion are used in conjunction to identify recent consolidations.

A particular pattern was used as opposed to random stock price data to provide a level of consistency between the stock charts used in training and testing. Ideally, this reduces the amount of noise produced by extraneous variables and provides a more directed approach to test the potential efficacy of the proposed algorithm. 211 stock charts of unique ticker symbols were detected by the scanning algorithm and used in the model.

The buy point for each of the detected patterns was coded to be the 'high' of the recent consolidation. The matplotlib web.DataReader() function was used to create the stock price daily chart images using the inputted stock ticker, start date of the chart, and the end date. For the purposes of the study, the start date was defined as four months before the buy date (the date that the stock price hit the buy point). Four months was chosen for the timeframe through trial and error as it seemed to provide enough data for the CNN network to identify patterns out of. The end date was defined as the date the buy price was hit.

In the preprocessing stage, the stock price charts were standardized with the open price for the first shown day being labelled as 1$ and all future prices subsequently given values relative to it. Extraneous visuals such as volume were removed, leaving only the candlestick price pattern for each stock where the top of each candle is the day's high, the bottom of each candle stick is the low, and the top and bottom of the candle colored bars are the day's open and close respectively. The colors of each candlestick represent the price change from that day's price close to the previous price close (green represents a price increase between the

closes, red: a price decrease). An example of the resulting stock chart image is shown in Figure
4.

The return of the stock, for simplification purposes, is defined as the percentage difference between the buy point and the close of the day 28 days from the buydate. Each stock chart is assigned a class label according to its return value.

Two cases were tested in this study, a binary classification (positive/negative) approach and a multi-classification approach (positive, negative, and neutral). The reason for testing on both model types was to start from the simplest generalization, whether a stock would move lower or higher. Then in the second approach, we attempt to eliminate insignificant returns (close to 0) using a neutral category. For the sake of this study, to create a relatively even number of charts per class in the multi-classification case, the stocks with a four week return of less than -4% were assigned to the 'Negative' class. The stocks with a four week return of greater than +4% were labelled as 'Positive' and all other stocks were labelled 'Neutral'. In both cases, the training case inputs for the proposed convolutional neural network were stock price charts as described above with their associated return class labels as the target variables.

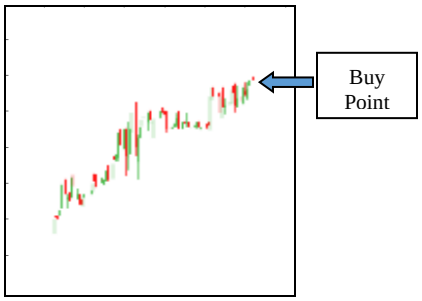


Figure 4. An example of a stock price chart image after preprocessing. The above chart had a return less than -4% at the end of the 4 week period following the last price day.

3. EXPERIMENTAL SETUP

In both the binary and multi-classification approaches, a roughly 80/20 training/testing splits were used. In both cases, the number of testing samples was kept constant across all classes.

Categorical crossentropy was used as the loss function for the multi-classification problem, while binary cross entropy was used for the binary approach, as we looked to maximize the validation accuracy of the testing set predictions to the actual return classes in each problem.

The architecture of the proposed convolutional network was optimized through an extensive hyper parameter search. The model used for both approaches consists of three 3x3 convolutional relu-activation layers with 2x2 max pooling sequences followed by flatten, dropout, and dense layers respectively. Each image is converted to a (150, 150, 3) shape for input into the first convolutional layer. The last dense layer uses a softmax activation to output the probabilities of the input image belonging to each class (sigmoid for binary case). The full architecture is shown in Figure 5.

No data augmentation techniques were used to increase the number of variations of images in the training datasets in order to maintain both horizontal and vertical symmetry in each input image. Since the basis of using this pattern in particular was to train on stocks with specific price sequences in a time-series format, such data augmentation would likely distort the relationships between these sequences.

The performance of the CNN models were compared to the VGG16 pre-trained convnet to determine the distinctiveness of stock price charts as a domain and to compare results to. Additionally, visualization techniques were implemented to observe what each activation channel may be detecting in the input images.

Layer (type)	Output Shape	Param #
conv2d_63 (Conv2D)	(None, 148, 148, 8)	224
max_pooling2d_63 (MaxPooling	(None, 74, 74, 8)	0
conv2d_64 (Conv2D)	(None, 72, 72, 16)	1168
max_pooling2d_64 (MaxPooling	(None, 36, 36, 16)	0
conv2d_65 (Conv2D)	(None, 34, 34, 16)	2320
max_pooling2d_65 (MaxPooling	(None, 17, 17, 16)	0
flatten_24 (Flatten)	(None, 4624)	0
dense_40 (Dense)	(None, 128)	592000
dense_41 (Dense)	(None, 3)	387

Total params: 596,099
Trainable params: 596,099
Non-trainable params: 0

Figure 5. The architecture for the proposed CNN is shown.

5. EVALUATION RESULTS

The proposed algorithm proved to accurately identify correct return classes 76% of the time on average for the binary classification test and 49% of the time for the multi-classification test. The performance comparisons between the proposed algorithm and the VGG16 pre-trained convnet are shown in Table 1. On the given training and testing data, the model showed to outperform the VGG16 model in both cases. The VGG16 model had resulting validation accuracies similar to what would be expected with random predictions (slightly better in the binary classification case). This in combination with the fact that the training accuracies of the VGG did not improve over time leads us to believe the pre-trained image weights of the VGG16 weren't adequately equipped to be applied to such a distinct domain

as stock image charts. The training and validation loss/accuracy of the models are shown over the course of 20 epochs in Figures 6 and 7 for the binary and multi-classification cases respectively.

Approach	Validation Acc. Binary	Validation Acc. Multi-Class
Proposed	0.76	0.49
VGG16	0.55	0.33

Table 1. The validation accuracies of the proposed method and the pre-trained VGG16 convnet in both the binary classification and multi-classification cases. Each accuracy value is the average of the last five epochs.

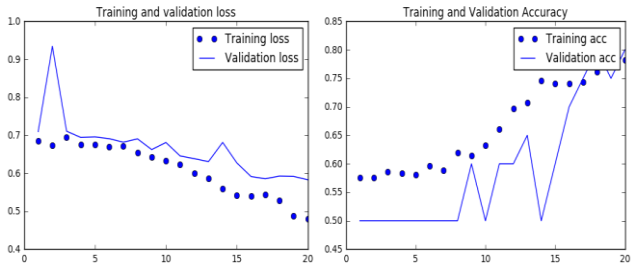


Figure 6. The training and validation losses and accuracies over 20 epochs of the implemented CNN in the binary classification case.

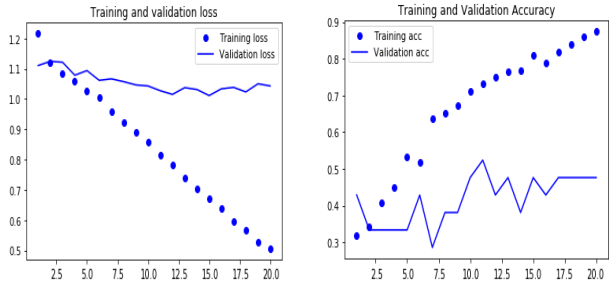


Figure 7. The training and validation losses and accuracies over 20 epochs of the implemented CNN in the multi-classification case.

In Figure 8, we can see an example of a stock price chart correctly classified by the algorithm to have a return greater than 4% with an assigned probability of 65%.

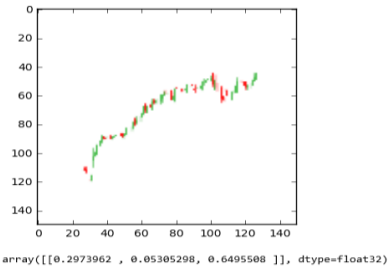


Figure 8. An example of a price chart of an inputted stock, with its associated assigned probabilities of being negative, neutral, or positive respectively.

The accompanying visualizations below were all extracted from the multi-classification model version. In Figure 9 we can observe what the fourth channel of the first convolutional layer is potentially picking up on in the stock chart shown in Figure 8. In this case, it appears the fourth channel of the ReLU activation is detecting the days with large price decreases from the day before (indicated by the large red bars in the original image). This is interesting both in terms of the CNN's understanding of the images and the potential real-life applications. It shows that the CNN is able to pick up on both the magnitude of the price bars as well as the colors simultaneously in the activation layer's channel. Relating this to its investing applications, big down days are used across the trading industry as signals to buy or sell.

Figure 10 shows the activations for all channels for the first and last convolutional layer for the stock chart in Figure 5. It can be observed that the activations appear more abstract in the last layer shown in (b) compared to the first layer where all filters are activated by the input image. The last layer isn't shown in entirety for space purposes, but we can also observe more blank filters in the last layer which implies the pattern produced by the filter isn't found in the input image. Finally, in Figure 11, we can see the class activation heat map passed into the last convolution layer when the CNN is tested on the Figure 8 image. Interestingly enough, the price consolidation at the end of the chart is identified as predictive in some way as relating to its 'Neutral' class prediction.

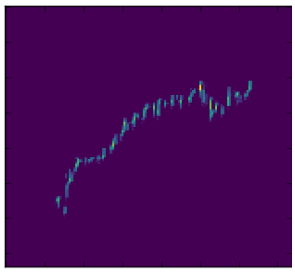
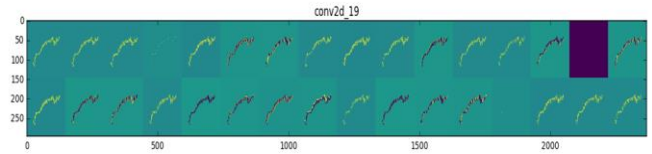
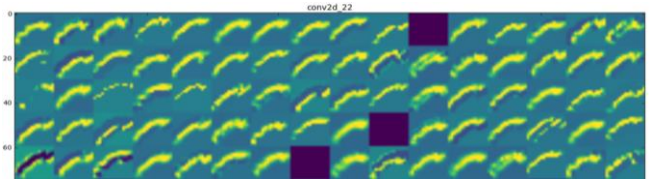


Figure 9. The fourth channel of the activation of the first layer of the CNN is shown, appearing to capture big down (red) days.



10 (a)



10 (b)

Figure 10 (a) (b). The ReLU activations for the first convolutional layer are shown for all channels in (a). (b) shows the activations for a subset of channels in the last convolutional layer.

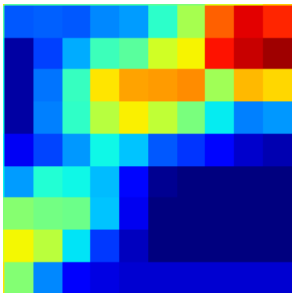


Figure 11. The class activation heat map uses the gradient information for the given class (here: neutral) passed into the last convolution layer and it highlights important class-specific regions

6. CONCLUSION

This work is the first found to investigate the potential application of deep learning vision analysis on stock price charts to investment strategy. The results showed the model to work effectively, in this respect, to classify stock charts as having a positive, negative, or neutral return from the buy point to four weeks in the future. Whereas in most cases, a ~75% binary accuracy and ~50% multi-classification accuracy would not be viewed as useful, in the case of trading, the resulting accuracies would be easily adaptable to creating effective trading systems.

Additionally, since accuracy in this sense is based only on whether the model's highest output probability matched the correct class label, the metric disregards the information value added from the probability distribution outputted. For example, if the model predicted from a stock chart that its future return has a 50% chance of being 'Positive' and a 45% chance of being 'Neutral', even if the end result was 'Neutral' and this classification was 'incorrect', a trader would likely be more inclined to purchase more shares since the predicted probability of a negative return is low, making the outputted probability distribution very useful in investment decision- making.

For simplification purposes, the models used in this project utilized classification approaches to create a hard-coded separation of the presented classes. This was done in an effort to create a simple translation to forming a trading system. However, in future work, I would be interested in testing an ordinal regression approach to predicting a future stock price direction and volatility to see if there's more value gained compared to the multi-classification method in terms of outputting an expected return and level of volatility. I would expect an ordinal regression framework to provide more information, especially in context of creating a more calculated trading strategy.

In terms of other modifications to the presented approach for future work, there are many variables that could be further optimized. For example, only time frames of four months and class division percentages of < -4% and >4% were used in this study for multi-class separation. Ideally, with more data available, different separations and time frames could be further optimized. Additionally, more indicators could be added instead of just price. It would be interesting to see if an adding an indicator like volume would add any additional predictive value. Since the goal of the approach is to form successful trading systems, instead of defining the target return value as the four week return, a more robust approach might also include making the return a profit target or percentage drawdown from a day's high after the buy point (ex: 10% from a set high).

While the results in this study are promising, only 211 cases were tested on. Testing on more data is needed to improve confidence in the model's efficacy. Here, only a very specific price pattern was trained and test on, thus creating a sparsity of data to choose from. I would be interested to see in future work with more data if chart timeframes with random patterns would have comparable results, to be able to ascertain whether a vision approach to stock trading would be effective in general, only with certain price patterns, or not at all.

11. ACKNOWLEDGEMENTS

I would like to thank Professor Kapil Thadani, Professor Liangliang Cao, and Professor Xiaodong Cui at Columbia University for their helpful feedback throughout the course of this project.

12. REFERENCES

[1] M. Velay and F. Daniel, "Stock pattern recognition with deep learning," August 2018.