Using Stock Charts to Predict Next Day(s) Close

Joel Riesen

Northwestern University MS Data Science

March 14, 2021

Abstract

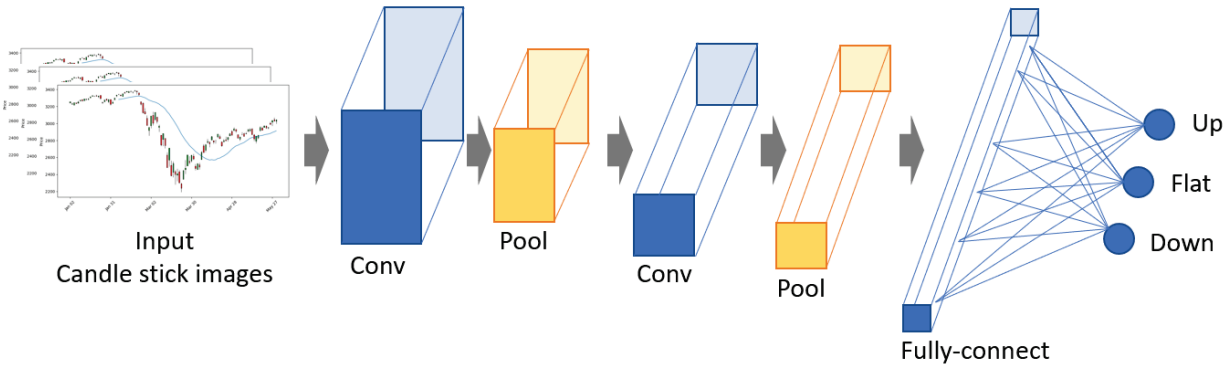
The main goal of this paper is to find out if it is possible to get a 90% accuracy on the close for the S&P 500 index. From the abundance of research papers out there, it seems like it should be possible. However, using those strategies in the real world is not easy. To achieve this goal, I am going to keep the training simple. I will be using stock graphs, not spreadsheets with Convolutional Neural Network(CNN). When a CNN breaks down its pictures, it does so to find the essential parts of the image (convolution). It's this process in which I believe CNN can see a pattern.

Introduction

Is it possible to achieve a 90% accuracy of a positive or negative close for the S&P500 next day? Not trying to get the price (yet anyway). This approach is that the market has several repeating patterns that have been well documented, for example. Thomas Bukowski (2)has written several books in which he has recorded the vast majority of the chart patterns and their historical performance. Thomas Bulkowski's website has a lot more information on stock chart patterns and their performance. Also, more details on his books, (1) http://thepatternsite.com/. A significant contributor to this strategy is from Benoit Mandlebrot's work with fractals and Brownian motion for emulating stock performance(2) (2b), which is discussed in more detail below.

Literature Review:

Using AI to predict stock prices has become an active topic for investors to make substantial returns. However, finding papers written using stock chart images for the data was not as prevalent as I thought it would be. Yohei Komori, "Convolutional Neural Network for Stock Price Prediction Using Transfer Learning,"(3) used stock images. The study also used a CNN with the pre-trained model weights from Keras - Inception v3.



"Convolutional neural network for stock price prediction using transfer learning" Yohei Komori page 3

Instead of just using a buy/sell ideology, they used an 'up,' 'down' and 'flat' as the output and gauge performance. The author took an interesting approach to train the data set. The author divided the data set into 30 bars and used a 10-day sliding window; 20 days are shared. So the testing window was

moved ten days instead of just one. I point this out to show that the chart sequence has a gap. A 25-day moving average was also used. The model used for the one-day forecast using the last day's closing price with the next day's close for training. When the next day's close was up or down 0.5%, it was labeled as up or down. If not, it was then labeled flat. The data was then run through the Inception v3 model with three additional layers. For the optimizer, the author used RMSprop the loss function was categorical cross-entropy. Two strategies were also documented along with this, Momentum and a contrarian strategy. The momentum strategy contained three parts

1. The last two days were up
2. The most recent close was the lowest close for the past 30 days.
3. The price closed over the 25 DMA

For "down," the momentum strategy would use the opposite of the three rules. It would be "flat" when neither condition is met. The Contrarian just used two conditions

1. The last two days were down
2. The most recent close was the highest close for the past 30 days.

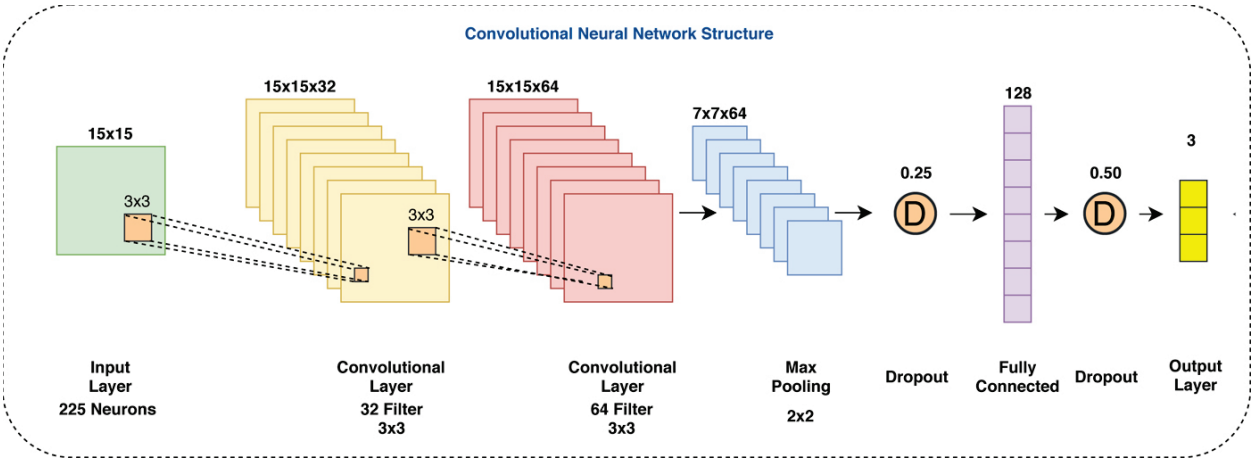
The performance for this Inception V3 was about 45%, both Momentum and Contrarian were below 30%.

Sim, Kim, and Ahn (2019)(4) looked at one-month-long one-minute bars or 41,250 minutes. Thirty-three thousand minutes were allocated for the training dataset leaving the 8,250 for testing, or 80/20% train/test. The data was converted to 30-minute images; the training data consisted of 1,100 images, testing was 275 input images. It ran a few tests using CNN with various inputs. They were:

1. CNN1 – Closing price
2. CNN2 – Closing Price, SMA, EMA
3. CNN3 – closing price SMA, EMA, ROC, MACD
4. CNN4 – Closing Price SMA, EMA, ROC, MaCD, Fast %K, Slow %D, BB Bands

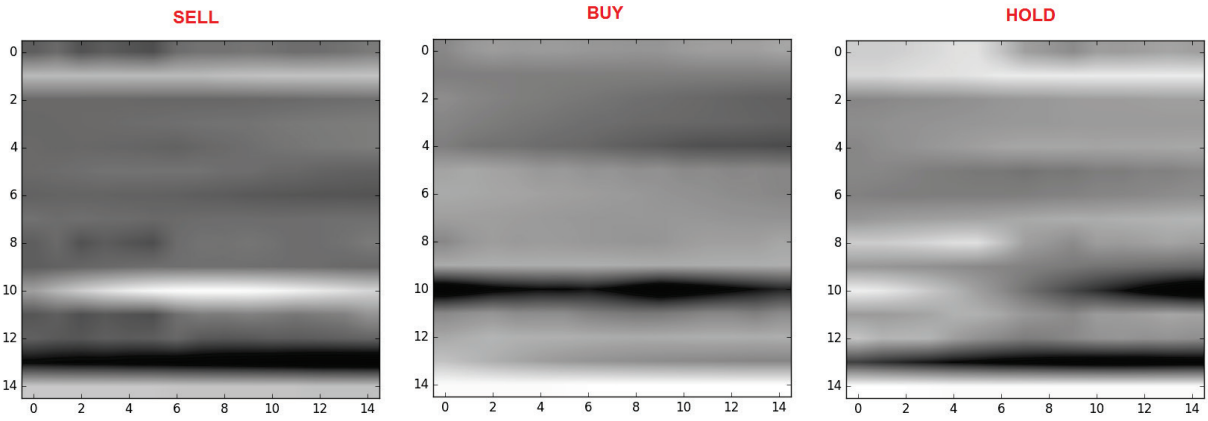
To evaluate their test, they used three measurements: hit ratio, sensitivity, and specificity. The hit ratio is a measure of its accuracy in predicting correctly. Using the SGD and ADAM optimizer, the highest hit accuracy was CNN1 with over 60%, with all the others > 60%, which left the authors stating that technical indicators cannot affect the positive impact of stock price forecasting using a CNN. This is why they concluded that using other factors that move opposite the stock price would help, like gold and interest rate. I would think using investor sentiment might be another idea to try.

Sezer and Ozbayoglu's (2019)(5) is another paper that used images with a CNN. They used data from 2002 to 2017 and used every five years for training, then one year for testing. They also used a Buy, sell and hold methodology. The indicators that they utilized for the test were: RSI, Williams %R, WMA, EMA, SMA, HMA, Triple EMA, CCI, CMO, MACD, PPO, ROC, CMFI, DMI, and PSI. They used an 11-day window in getting their data from the charts. For the CNN, they used several different layers: convolutional, max-pooling, dropout, and a whole connected MLP layer.



"Algorithmic Financial Trading with Deep Convolutional Neural Networks: Time Series to Image Conversion Approach": Omer Berat Sezera, Ahmet Murat Ozbayoglua page 12

This CNN was fascinating to me since they broke the images down and forced the AI to find the essential features to move on. An example from an image of their charts after going through the CNN

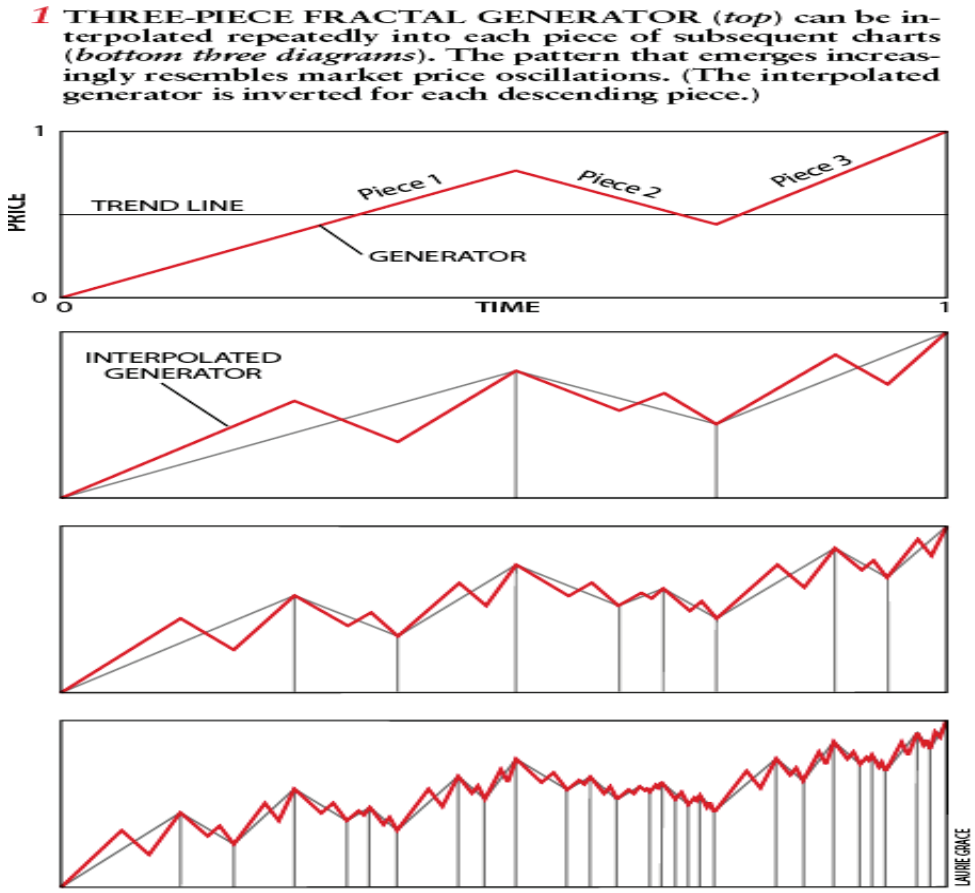


"Algorithmic Financial Trading with Deep Convolutional Neural Networks: Time Series to Image Conversion Approach": Omer Berat Sezera, Ahmet Murat Ozbayoglua page 12

I find the horizontal line at the number 10 on the images to be interesting. Even on the Hold image, the line is between black and white. Some have suggested using autocorrelation first to see if it improves performance by getting rid of some extra noise.

In backtesting, the CNN performance held up and beat a few other strategies, including buy and hold. I thought the LSTM and CNN would have a similar performance. The average and sd for CNN are comparably good too.

The article *In Homage to Benoit Mandlebrot*, by Jonathan Scott(6). He discusses the use of Benoit's fractal geometry in the financial markets and how to emulate them—using a Brownian motion in multifactual time, which first starts as an initiator as a straight line. The generator, a lightning bolt, or an elongated z shape would theoretically be placed over the straight line in an up-down-up fashion, which can be repeated repeatedly. When finished, it will look like a stock chart. The paper also discussed another important factor, time. In which Mandelbrot observed that markets moved in fast and slow periods. To me, that is when the market has a higher volume than usual, which you can see when using bars that are not time-based, like Renko, range, tick, and even point and figure(PNF). Using the Holder or Hurst exponent, one can gauge if the market is standard, persistent, or anti-persistent(7). Another way to put it is a random walk (H ~ 0.5), trending (H > 0.5), or mean-reverting (H < 0.5) for a specific period.



1 THREE-PIECE FRACTAL GENERATOR (*top*) can be interpolated repeatedly into each piece of subsequent charts (*bottom three diagrams*). The pattern that emerges increasingly resembles market price oscillations. (The interpolated generator is inverted for each descending piece.)

https://www.scientificamerican.com/article/multifractals-explain-wall-street/ By Benoit B. Mandelbrot on September 15, 2008

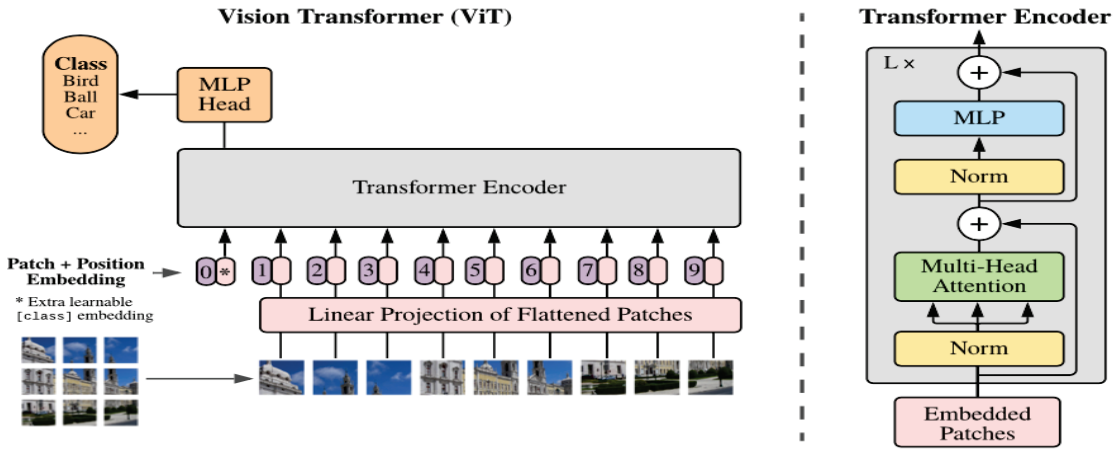
Data

The primary data for this is the price history for the stocks to convert into the stock charts. I hoped that I would come across a script or a method to help with image conversion data. However, most of them are very specific to their papers. As for the size of the images, I would have to say that I would like to use the

nine-layered CNN from Sezer and Ozbayoglu (2019). Since I believe that is where most of their performance came from. Also, since I am looking for the fractals or the repeating patterns, it would be a lot easier to find them using continuous data instead of being split every 10, 15, or 30 bars. Also, since fractals move in their own trading time. (Mandelbrot) I will also be using Renko bars. Renko bars can be set at a determined amount. Instead of having the price bar move with every tick, it only moves at a set amount. Renko bars do not take time into account; it plots the next bar, whether it is up three or down three. This means certain days could have bars and others none. Here is an example of a three Renko price bar compared to a six.

Research Design and Modeling Method(s):

I would have to start with the nine layered CNN from Sezer and Ozbayoglu's (2019) paper. I think that the main "gains" for accuracy and performance would be sticking with a CNN in some form but maybe add the layers differently, perhaps with an autoencoder? Another model that I think I would like to try would be the transformer from google, the Vision Transformer (ViT)



The ViT uses the input image as a sequence of image patches and directly predicts class labels for the image. The ViT does need a lot of data to be accurate. (8)

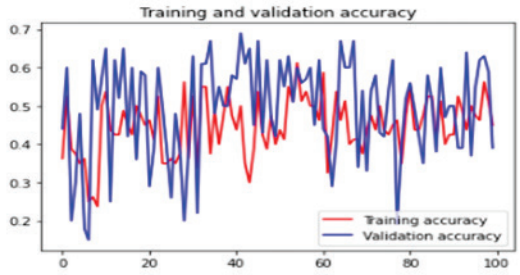
Design and Implementation Considerations:

Having all these different charts is a lot of data. I have started to use an external hard drive to place the stock charts for storage. Then I do not have to download the data, create the charts then run the AI. I want to start saving intraday data. Using an external hard drive makes it easier to work on different computers as well.

Results:

For these results, I ran the CNN from Yoheiko paper "Convolutional neural network for stock price prediction using transfer learning"* The difference between the two was that I used graphs with just 5 bars and a two-period simple moving average. I found the difference between the two very exciting.

Original method:




```
<Figure size 432x288 with 0 Axes>
```

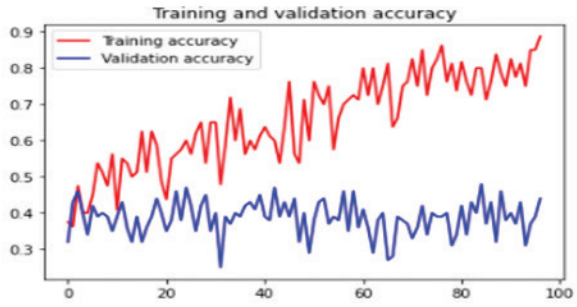
How can it be this bad?

```
In [27]:
1 print("Evaluate")
2 result = model.evaluate(train_generator)
3 dict(zip(model.metrics_names, result))

Evaluate
589/589 [==============================] - 92s 156ms/step - loss: 1.0173 - acc: 0.5034

Out[27]: {'loss': 1.0173472166061401, 'acc': 0.5033955574035645}
```

Smaller sequenced charts:




```
<Figure size 432x288 with 0 Axes>
```

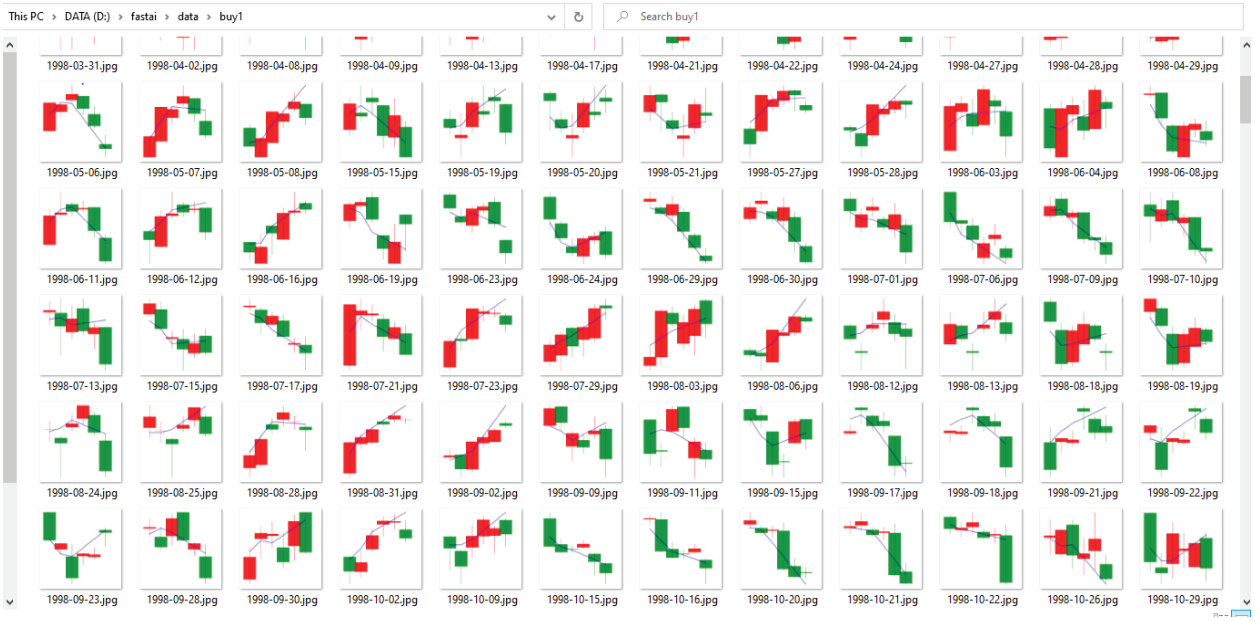
How can it be this bad?

```
In [14]:
1 print("Evaluate")
2 result = model.evaluate(train_generator)
3 dict(zip(model.metrics_names, result))

Evaluate
122/122 [==============================] - 44s 359ms/step - loss: 0.3248 - acc: 0.9067

Out[14]: {'loss': 0.3248049318790436, 'acc': 0.9066886901855469}
```

These are the charts used:



More information is needed to finalize the results.

*AI that was done by Yohei Komori from the paper "Convolutional Neural Network for Stock Price Prediction Using Transfer Learning." (https://github.com/YoheiKo/CNN-candle-stick).

Analysis and Interpretation:

I need a lot more information, but I was thrilled to see that smaller charts worked. The training accuracy was getting very close to the 90% I am trying to achieve.

Conclusions:

Need more information

Directions for future work

(placeholder)

Next-frame prediction with Conv-LSTM from Keras. I think that the Renko bars will be a great tool to use.

https://keras.io/examples/vision/conv_lstm/

Appendices.

1. Bulkowski, Thomas N. ThePatternSite.com. Accessed March 17, 2021. http://thepatternsite.com/.

2. Mandelbrot, Benoit B., and Richard L. Hudson. *The (Mis)Behavior of Markets: a Fractal View of Risk, Ruin, and Reward*. London: Profile, 2008.

2b.Mandelbrot, Benoit B. "How Fractals Can Explain What's Wrong with Wall Street." Scientific American. Scientific American, September 15, 2008. https://www.scientificamerican.com/article/multifractals-explain-wall-street/

3. Komori, Yohei. "Convolutional Neural Network for Stock Price Prediction Using Transfer Learning." SSRN, December 31, 2020. https://ssrn.com/abstract=3756702 .

4.Is Deep Learning for Image Recognition Applicable to Stock Market Prediction? Hyun Sik Sim , Hae In Kim, Jae Joon Ahn Published 19 February 2019

5. Algorithmic Financial Trading with Deep Convolutional Neural Networks:Time Series to Image Conversion Approach Omer Berat Sezer, Murat Ozbayoglu

6.Scott, Jonathon. “In Homage to Benoit Mandelbrot.” Towards AI - The Best of Tech, Science, and Engineering, December 23, 2020. https://towardsai.net/p/mathematics/in-homage-to-benoit-mandelbrot .

7. Dudia, Ashwin, Vivek Kumar, and Ritabrata Bhattacharyya. "Short Term Trading Model for Asian Equity Index Futures – Using Hurst Exponent." SSRN, March 23, 2020. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3543079 .

8. Houlsby, Neil, and Dirk Weissenborn. "Transformers for Image Recognition at Scale." Google AI Blog, December 3, 2020. https://ai.googleblog.com/2020/12/transformers-for-image-recognition-at.html.

Cohen, Naftali. Trading via Image Classification. https://arxiv.org/, October 26, 2020. https://arxiv.org/pdf/1907.10046.pdf .

"Random Fractals and the Stock Market." Fractal Geometry. Accessed March 14, 2021. https://users.math.yale.edu/public_html/People/frame/Fractals/RandFrac/Market/TradingTime/TradingTime.html .