

Miled Ismaël | Kschool Master Deep Learning | 2019-2020

TFM Introduction to Deep Learning Tools For Finance

Application to Transfert Learning for Technical analysis

# Introduction

The objective of this project was implement the methodologies of deep learning on my day-to-day work. Some gave interesting results and were quite straight-forward to implement but some still need some extra push.

The objective of the transposition of deep learning methodology to finance was to focus on:

1. classification of current/future of market/economic state
2. prediction of future price
3. construction of investment strategies
4. risk estimation
5. process news flow to assess market sentiment

For instance, we can focus on Reinforcement Learning to improve the investment strategy as some market can be viewed as massive multiplayer game and we could have better estimation of hidden risk replacing Monte Carlo methods by Generative adversarial network.

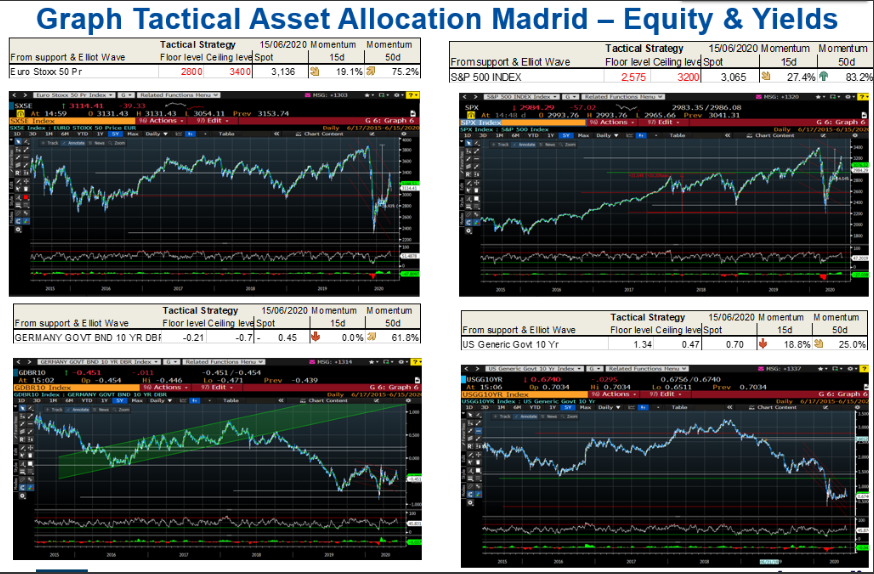
So in this case I have worked and implemented Auto encoder for market state classification, LSTM for price prediction, GRU, Seq2Seq, RNN, CNN, Convolutional Bayesian NN on financial and macroeconomic data. Those models still need to be improved but as a starting point, it raises some hope.

In any case, for this TFM I will focus deeper only on one specific problem where we will try to answer the question of the real validity of Technical Analysis which a methodology for forecasting the direction of prices through the study of past market data, primarily price and volume. The ground of this theory comes from behavioral economics and quantitative analysis, which stands in contradiction of much of modern portfolio theory and market valuation.

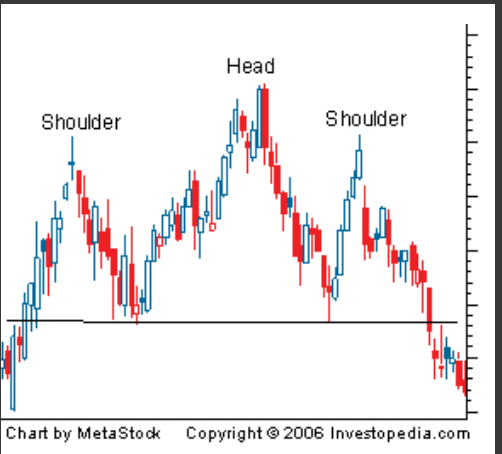
Moreover, as many operator apply the same strategy some psychological anchors can be set and are hard to break and contradict. I consider here a problem of behavioral finance as every investor look thoroughly at those graphs more than fundamentals numbers. Just because the human brain can much easier interpret a graph than figures it is the same reason that he has more emotion toward a music melody, which is succession of notes than a lonely FA. We can note that those graph can be interpreted on small (minutes) or long (years) time frame so as to get an estimation of its evolution so it can be extended the investment decision methodology for any time horizon. The human brain process this information deeply and the consequence of this process is the main behavior of stock market.

Benjamin Graham in the "Intelligent Investor" - written in 1949 and considered as the bible of value investing - introduce the allegory of Mr. Market, meant to personify the irrationality and group-think of the stock market. As of august 2020, the value of some stocks are higher than the economy of France and Germany and some companies (Tesla) are bought at a price quite difficult to apprehend in terms of valuation fundamentals and comparison to established company in Europe (Volkswagen) or Japan (Toyota). That is true, that our brain is set to always find an explanation but in this approach we' ll try to apprehend the impact of price evolution that makes Mr Market more greedy or fearful.

Example of tactical asset allocation based on Technical analysis . source amundi and bloomberg



Example of technical analysis signal : Head and shoulder



# Data Description Recuperation and Exploration

For this part I focused on building my own dataset and thus it was quite time consuming and many algorithms need to be improved.

The ideas for financial technical analysis is a graph of price evolution. Thereafter we can focus on adding the features like open close high low and in another unit volume. Many traders or finance scholars tried to add more complex formula like momentum, RSI, Bollinger Band in order to justify its investment decision. But in any case the base remains the price evolution through a simple graph.

Also I selected the American equity index SPX 500 for this project as it is the most liquid equity indices where many operators can interact from retail to institutional investors. Moreover the index has a long history that I see from 1927. Also the index price are quoted every working day.

In terms of information source, I had the great range of provider but I chose the yahoo finance API source so as to download the data. The datas downloaded are for working day 6 elements open close high low known better as OHLC, Adjusted Close (for dividend, spin of etc) and volume (typically the number of contract to buy or sell exchange during the day).

As we are focusing on graph I have chosen a 25 days horizon data for drahe past and 5 days for the future. For the past is typically what the standard technical analysis formula stand for 1 month of market. And for the future 5 days fit in one calendar week. In any case it is a choice.

For the graph as an image, which are my X\_train and Y\_test and in my dataset I have decided to take a figure created from matplotlib library and convert it to a np array and adjust the size to (32 ,32, 3) so as to fit the input of the neural network. NB: Although it is good so far to start I am currently modifying the algorithm to write and resize the value into a (N,N,N,3) np.array to get the OHLC and volume level of the day.

For the Y\_train\_StateClass and Y\_test\_StateClass I have chosen to split the future in 5 market state plus 1 for the error case (when we have no 5 days forward future )

Image X Y\_StateClass time

**BB**



**SN**

SS

**N**

**NB**

Y State Class 0: Sell-Sell | 1: Sell- Neutral | 2: Neutral | 3: Neutral -Buy | 4: Buy -Buy |5: Error

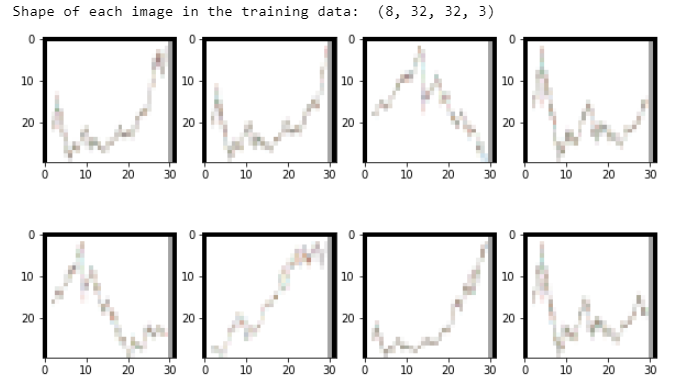
We have worked on times series dataframe therfore we needed to flatten the image X and get it back to a (32, 32, 3) format for the entry of the Neural Network. We choose to normalise the value of the image X to a value between 0 to 1. We normalise also the value of sp500 divising by it s maximum historical value. the testing and training time series dataset are shuffled by the date of reference with a split number of 0.8.

At the end I splited the datas in 4 np.array group x\_train, x\_test and y\_train, y\_test owith a total of 23254 image of 32 x 32 x 3 and 23254 state

Please note that:

* I had to modify the algorithm with figure matplot lib so as to avoid consuming the RAM and crash the system
* We can increase the dataset taking into account the evolution very liquid stocks or other indices as long as we have very high liquidity and number of participants
* The calculation of the dataset can take more than 8 hours of calculation as the code is not optimized so far as said we can quickly implement parallel computing and rapid image setup instead of using matplotlib library
* Google colab calulcation speed decrease after some time typically initially it calculate 2 steps per second and after 5hours we have 1 step every 3 seconds so I launched several colab with smaller disjoint steps and merge the results therafter.

The S&P 500 is a stock market index that tracks the stocks of 500 large-cap U.S. companies. It represents the stock market's performance by reporting the risks and returns of the biggest companies. Investors use it as the benchmark of the overall market, to which all other investments are compared.



# Description of methodology and tools used

For the hardware I worked on google colab but it has some limitations in terms of time consumed but it permits to get access on a CUDA envireonment quite easily. I have tried in my local envirmonement and in gcloud but I had some difficulties to install the right version of the packages. I am trying at the same time to replicate the colab environment via Docker without any success so far.

For this work I have chosen to inspire me from the transfer learning of convolutional neural network example we studied for mnist and the fashion dataset.

In our case I use the Vgg1 16 model with additional layers to train so we have 66438 parameters to fit. In order to train, as I still have some difficulties to get a decent accuracy I launched a process to change different parameters (batch size, epochs, learning rare and optimize ) and in order to find the best set.

I have used also some callbacks so as to save the best current model for each epoch and if colab crashes I could take back my latest model for each try.

For the neural network library I have chosen tensorflow.keras as the package for image is quite complete and straightforward.

As we have a problem of classification I have used the loss measure of catgorical cross entropy and the list of parameters checked are the following:

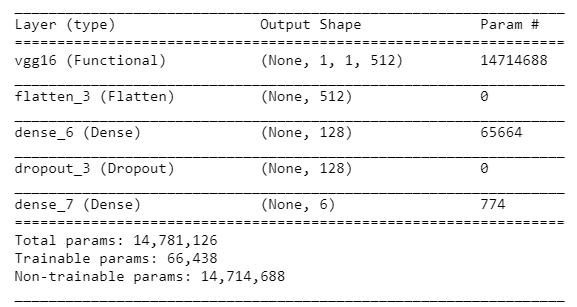
batch\_size=[25,50,100]

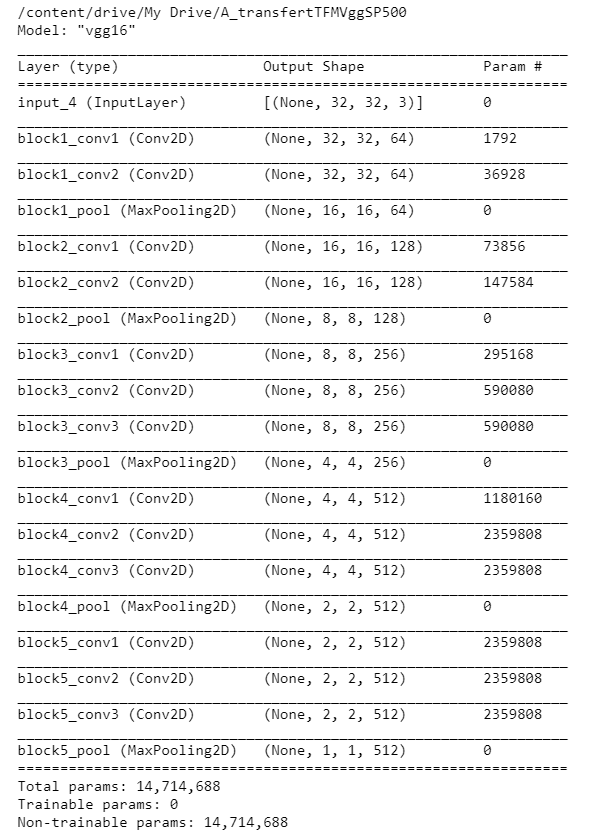
epochs=[25,50,100]

learning\_rate=[0.001,0.01,0.1]

optimizer\_name= SGD, RMSprop, Adam, Adagrad, Adamax, Ftrl

I have also tested the version of a learning rate with exponential decay

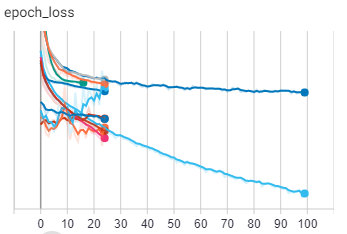
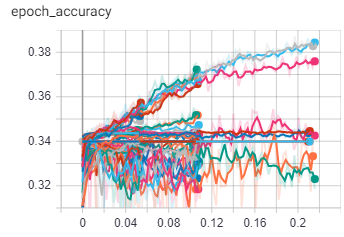


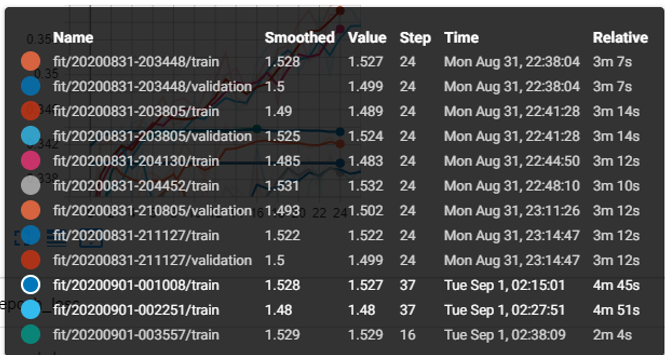


# Summary of the current results

So far the results are quite below expectation returning an accuracy of 38% and around 1.5 of cross entropy loss.

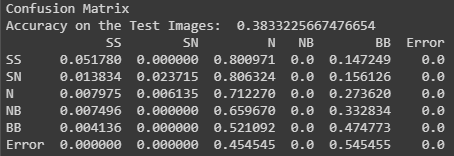
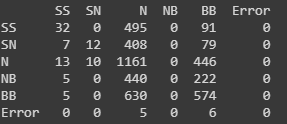
Here is the results so far with a comparison of the model with different parameters

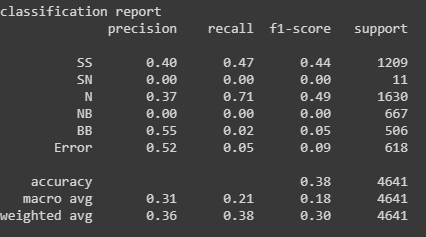




This part will evaluate the model with the testing dataset that we generated in first step.

We show the accuracy, the confusion matrix and the classification report of the best model for the test dataset. We notice that there is no N-B result so that needs to be investigated in terms of dataset.



# How to use it: Guess future market state from stock graph

The objective with a well-calibrated model with a high accuracy is to take an image of an historical graph from a market webpage like investing.com and save it to the the folder here ImageM/ with name image1.PNG or you can change the value of image path to the link you need.

This execution of the step 4 python command will answer the most appropriate decision for a time horizon of 1/5 (5day/25day) of the historical time for the graph. I we consider that we have a fractal behaviour of the stock market process which we suggest that its logarithm follow a Brownian process.

In clear it tells us which market state in the future is the best representative future market state for the image and we have to buy or sell or keep neutral.

# Conclusion

The results of this study are a bit disappointing in terms of accuracy and confidence in the model maybe we can conclude that technical analysis has no self-fulfilling realization by the market and can be ranked close to astrology or we can say that some work to improve the algorithms and model is needed.

For instance, to improve the model but after checking all the parameters I will be focusing on

* get more dataset on stock (AAPL, MSFT) and indices (Nikkei, Eurostoxx, MSCI WORLD)
* train the whole vgg16 this time untrained
* apply to other CNN trained like AlexNet or Mask R-CNN
* increase the size of input image from (32,32,32,3) to (256,256,256,3)
* calibrate a generative adversarial network to increase dataset and self-fulfilling conclusion
* Improving the dataset of the image to include the candlestick and volume for the series,

NB: On the last week of work I found 2 studies which can be comparable this time using candlestick chart classification of current state with Hu Hu 2017 and Rosdyana Manguir 2018 which can be inspired because of the