# Stock Market Prediction using Artificial Neural Networks

##### **Fady Mohamed, Karim Mady**

\* Computer Science and Engineering, The American University in Cairo

**Abstract-** In the trading environment, high-quality one-step forecasting is usually of great concern to market makers for risk assessment and management. We aim to forecast the price movement of individual stocks, based only on their historical price information using artificial neural networks.

**Index Terms**- FNN: Fully Connected Neural Network

RNN: Recurrent Neural Network

CNN: Convolution Neural Network

GAN: Generative Adversarial Network

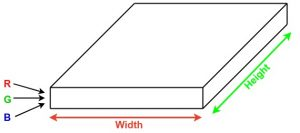
**Introduction**

Foreseeing stock costs is a vital target in the monetary world, since a sensibly exact predictions has the likelihood to yield high monetary advantages and fence against market dangers.

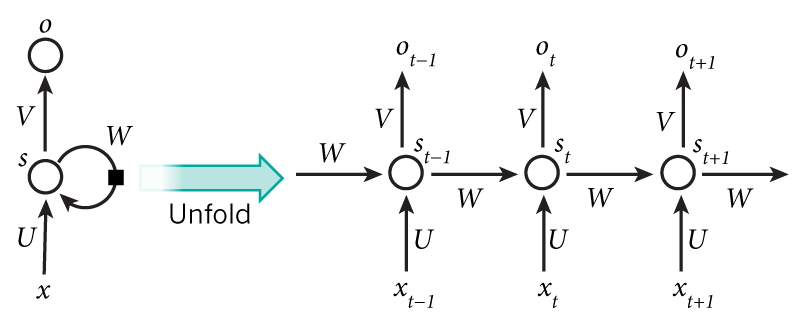
In most cases, the forecast results are surveyed from two angles: the first is estimate Error (mostly the RMSE (Root Mean Square Error) or RMSRE (Root Mean Square Relative Error)) between genuine cost and estimate value; the second is heading prediction precision, which implies the rate of right forecasts of price series direction, as upward and downward developments are the main thing for choice making. Indeed, even little enhancements in prescient execution can be entirely productive.

However, predicting stock costs isn't a simple work, due to the unpredictability and disorderly elements of the business sectors and the numerous non decidable, non stationary stochastic factors included. Numerous researchers from various regions have contemplated the authentic examples of budgetary time arrangement and have proposed different strategies at determining stock costs. So as to accomplish promising execution, the majority of these ways require watchful choice of info factors, setting up prescient model with expert money related information, and receiving different measurable strategies for exchange examination, which makes it troublesome for individuals outside the money related field to utilize these strategies to foresee stock costs.

Convolutional Neural Networks or covnets are neural networks that share their parameters. Imagine you have an image. It can be represented as a cuboid having its length, width (dimension of the image) and height (as image generally have red, green, and blue channels).



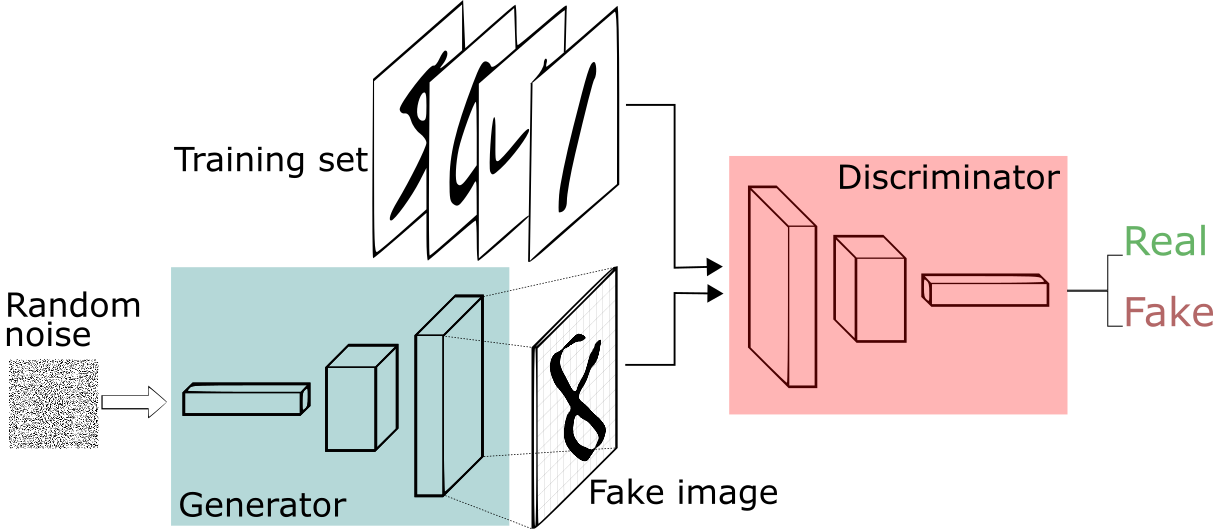
The idea behind RNNs is to make use of sequential information. In a traditional neural network we assume that all inputs (and outputs) are independent of each other. But for many tasks that’s a very bad idea. If you want to predict the next word in a sentence you better know which words came before it. RNNs are called *recurrent* because they perform the same task for every element of a sequence, with the output being depended on the previous computations. Another way to think about RNNs is that they have a “memory” which captures information about what has been calculated so far. In theory RNNs can make use of information in arbitrarily long sequences, but in practice they are limited to looking back only a few steps (more on this later). Here is what a typical RNN looks like:



GANs are generative models devised by [Goodfellow et al](https://arxiv.org/abs/1406.2661). in 2014. In a GAN setup, two differentiable functions, represented by neural networks, are locked in a game. The two players (the generator and the discriminator) have different roles in this framework.

The generator tries to produce data that come from some probability distribution. That would be you trying to reproduce the party’s tickets.

The discriminator acts like a judge. It gets to decide if the input comes from the generator or from the true training set. That would be the party’s security comparing your fake ticket with the true ticket to find flaws in your design.



**Existing Solutions**

One of the existing solutions is using Recurrent Neural Network alone in order to estimate the next time sequence using LSTM or GRU neural units. LSTM is a basic deep learning model and capable of learning long-term dependencies. A LSTM internal unit is composed of a cell, an input gate, an output gate, and a forget gate. LSTM internal units have hidden state augmented with nonlinear mechanisms to allow state to propagate without modification, be updated, or be reset, using simple learned gating functions. LSTM work tremendously well on various problems, such as natural language text compression, handwriting recognition, and electric load forecasting.

Another solution is creating images from the stock and feeding them into a forward Convolutional Neural Network. Which works like extracting important features from such images, and prediction the next time sequence according to the extracted

features

Let Xt represent a set of basic indicators and Yt denote the closing price of one stock for a one minute interval at time t (t = 1,2,...,T), where T is the maximum lag of time. Given the historical basic indicators information X ( X = {X 1 ,X 2 ,...,X T }) and the past closing price Y ( Y ={Y 1 , Y 2 ,...,Y T }), our goal is to predict the closing price Y T + 1 for the next one minute time interval.

**Proposed Solutions**

For the purpose of our experiment, We have two

solutions proposed that were not implemented

before. So we decided to run them versus, the

already implemented architectures, CNN, RNN.

The first proposed solution is to implement an architecture that can extract features from the sequence of inputs to the network, and then use the extracted features to predict the next time sequence based on a window of the extracted features.

The second proposed solution is to implement a Generative Adversarial Network.This model is based on the intuition that a trader predicts the upcoming sequence,and then he tries to verify and adjust his

prediction.

**Dataset**

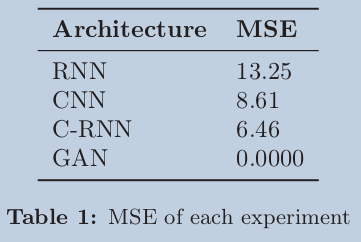
For the purpose if this experiment, we used the publicly available stock market dataset from Kaggle, which contains over 14 million stock market closing data point. However, we used only the 4 stocks of Google, Amazon, Facebook, and Netflix, as they have a closely related data points which would make the experiment run smoothly.

**Experimental Results**

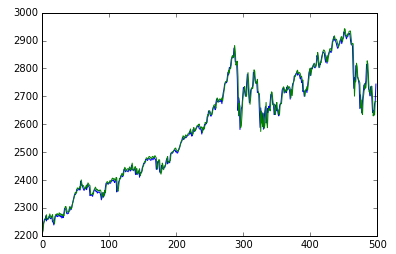
To ensure fairness in our experiment, we used the same dataset for all the models we trained. We tried several window sizes for the model. And, we finally choose the window size to be 8 for all models.



We used the MSE to be our loss function as we are comparing continuous values together.

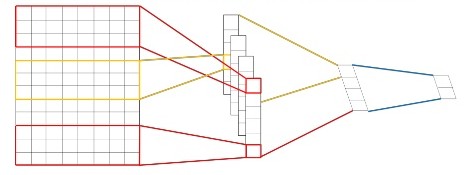


As you can see in the table the C-RNN gave us the best results . However we cant say that its the best implementation because we did not test the GAN completely, and we expect it to output better results then the C-RNN. In the graph below you can see the results of the C-RNN compared with the real values.

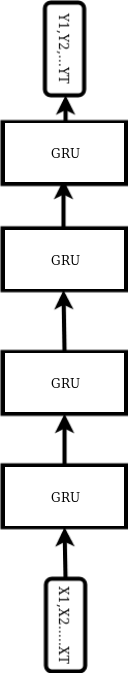


**Architectures Used**

For the CNN we used 2 stacks of ConvNets. Each stack consists of 2 size 3 filters to have a receptive field of 5 while having lower number of parameters and average pooling at the end as we are dealing with a sequence of numbers. We increase the number of filters as we go down the network to extract more features as we go along.

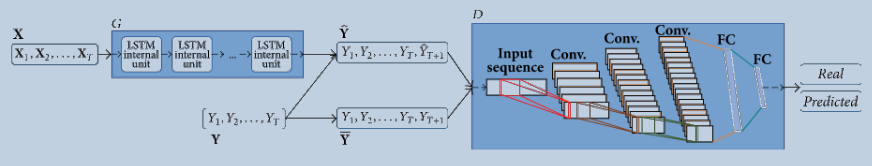


For the RNN we added a sequence of Recurrent Units. We ended up choosing the GRU units as they lead to faster convergence and performance.



For the C-RNN we combined both of the convolution to extract features from the sequence and then feed those features into the GRU units for past input dependence.

Our GAN architecture is as follows, we have our generator as a Recurrent Neural Network which takes the stock dataset as input and predicts on that sequence. We then train the discriminator based on the real dataset and the fake dataset. We then freeze the discriminator, and train the whole model on the real dataset.



**References**

Wang, Zhiguang & Oates, Tim. (2015). Imaging Time-Series to Improve Classification and Imputation.

Xingyu Zhou, Zhisong Pan, Guyu Hu, Siqi Tang, and Cheng Zhao, âĂIJStock Market Prediction on High-Frequency Data Using Generative Adversarial Nets, <https://doi.org/10.1155/2018/4907423>.

Mahdi Pakdaman Naeini, H. Taremian and Homa Baradaran Hashemi, "Stock market value prediction using neural networks," (CISIM), Krackow, 2010, pp. 132-136. [https://doi:10.1109/CISIM.2010.5643675](about:blank)

[arXiv:1805.11317](https://arxiv.org/abs/1805.11317)