# THE EFFECT OF GENETIC ALGORITHMS AND ATTENTION MODULES ON LONG SHORT TERM MEMORY NETWORKS FOR STOCK FORECASTING

ANISH NEERVANNAN [ANISHRN@SEAS], VINAY SENTHIL [VINAYKSK@SEAS], BHASKAR ABHIRAMAN [BHASKARA@SEAS],

ABSTRACT. In this report, we characterize the performance of long short term memory (LSTM) networks for stock forecasting and study how they are affected by attention modules and genetic algorithms. We find that due to the highly stochastic nature of daily stock prices, LSTM predictions are comparable to random guessing. In the long run, the LSTM predictions were able to accurately determine the general upward or downward trend of stock prices. Attention modules marginally improve this performance. Since genetic algorithms for updating weights ignore the gradient, they require significant training times in order to achieve useful changes in loss; nevertheless, due to the challenging domain of stock prediction, genetic algorithms could be a useful tool in tandem with traditional gradient descent methods.

## 1. INTRODUCTION AND BACKGROUND

Predicting how the stock market will move poses tough modeling challenges. With the behavioral, psychological, and irrational behavior of retail investors and traders constantly affecting prices, predicting the price movement of stocks even in the long-term is incredibly complex task. However, financial markets produce enormous amounts of data on a real-time basis, and one of the advantages of deep learning models is their ability to parse signals from large amounts of data.

Long Short Term Memory (LSTM) networks exhibit outstanding performance for predicting time-series sequences. This makes them a natural candidate for the prediction of stock prices, which are extraordinarily noisy and challenging to forecast. Attention-based mechanisms can augment LSTMs by learning how much weight to give previous samples in the time series. Another interesting approach in machine learning is the genetic algorithm (GA). GAs are (roughly) inspired by natural mechanisms of evolution. Populations of many randomly initialized models or sets of parameters are tested with a fitness function. Top performers are used to create new generations, whose attributes are perturbed by mutations (noising of weights/parameters) or crossing over (swapping attributes). When hyperparameter spaces are too large to sweep by brute force, GAs can be useful to tune them. In some scenarios, GAs can be useful for tuning the weights themselves; if the energy landscape of the loss function is not favorable for stochastic gradient descent, genetic algorithms provide an alternative.

There is a substantial body of research that has been conducted on stock prediction with machine learning, including LSTM models and attention modules. None have characterized the interplay between attention modules and updating weights via genetic algorithm in this setting; this is the objective of our experiments.

### 1.1. Contributions.

In this study we:

(1) Verify that attention modules improve the performance of LSTMs for predicting time-series stock price data.
(2) Show that genetic algorithms or other optimization approaches with many randomly instantiated parallel models could prove competitive in settings in which stochastic training data leads to craggly energy landscapes.

To view our work, we have linked our Github repository.

## 2. RELATED WORK

Due to the strength of the LSTM architecture for predicting time-series sequences and the ease of manipulating numerical data such as stocks, using LSTM for stock forecasting has been common in the literature. In 2015, Chen et al. used LSTMs trained on 30-day-long sequences to predict stock returns 14.3% to 27.2% better than random [1]. Though this improvement seems modest, beating out random predictions is profitable in high-volume, high-frequency trading scenarios. In 2017, training on Brazilian stock exchange data, Nelson et al. achieved an LSTM which predicts if a stock will go up with 55.9% accuracy, which is certainly enough to profit [2].

Attention mechanisms are a newer idea in machine learning, so the literature on LSTMs with attention modules for stock prediction is less popular. In 2017, Qin et al. used a dual-stage attention-based recurrent neural network to predict stocks on the NASDAQ 100 [3]. In their study, they used one input attention mechanism to learn input features based on the previous encoder hidden state, and another attention mechanism to select relevant encoder hidden states across all time steps. This two-phase method was able to outperform the state-of-the-art for time series prediction. In 2018, Liu et al. implemented an attention method to interpret both numerical stock data and news articles to predict stock performance [4]. The attention mechanism for news parsing was able to successfully filter the noise in order to take meaningful insights from the news in order to get an edge over single-source data streams.

Genetic algorithms have served some as an innovative way to augment the predictive power of neural networks. In a seminal work from 2000, Kim et al. used genetic algorithms to optimize weights and the discretization of the stock value inputs of a neural network with twelve inputs [5]. This GA used mutation, random noising of the weights inspired by DNA mutations, and crossover, swapping of weights inspired by chromosome section swaps during meiosis. In this work, the GA was used in a hybrid model with gradient descent to achieve optimal fitness functions, i.e. losses, over network weights and input feature discretization thresholds. This study found 10-11% improvement with this hybrid GA approach compared to networks trained on backpropagation or genetic algorithms alone. In a more modern work from 2018 by Chung et al., a GA was used to tune window size and topology for an LSTM for forecasting on the Chinese stock market [6]. By using a GA to search for the best network architecture, this team outperformed benchmark models.

In this report, we will investigate genetic algorithms for the purpose of tuning the actual weights of an LSTM rather than network hyper-parameters. Furthermore, we will see how genetic algorithm weight updates impact the attention module. Genetic algorithms for weight updates can be useful in situations where gradient information is sporadic, or the gradient is often zero, eg. reinforcement learning for videogames with binary reward functions. Though the LSTM gradient is straight forward to calculate in this study, we are interested to see if GAs will offer any benefit for the highly stochastic problem of predicting over stock datasets.

## 3. Approach

For our training data, we used daily opening prices of stocks on the New York Stock Exchange, NASDAQ, and NYSE MKT obtained at this kaggle.com page. These stock histories start from their IPOs and are logged until November 2017. We decided to focus on S&P 500 stocks to guarantee a decent history of stock prices. We also selected a sector to assist the model in identifying sector-based trends. For this study, we focused on the last 750 days of stocks within the information technology sector. This was split into a training/validation set with 600 days of stock history, and a test set of 150 days of stock history. We predicted on sequence lengths of 90 days, which was chosen specifically because 3 months is a common time frame for technical analysis and is the length of a quarter. We used mini-batches of size 8 for stochastic gradient descent. Before training, we standardized each stock to be within the range [-1, 1], so that the model focused on learning patterns in relative price changes rather than the absolute value of the price, which varied heavily across stocks. Since the output of our models was a real value, we used standard MAE loss, which is a standard choice for stock prediction problems.

The base LSTM model we used takes as input a time-series of variable length with an input feature of size 1, and outputs a hidden vector of size 100. Then a fully connected layer is applied to the hidden vector to convert the features to a singular prediction at each time step. The model's prediction is the price of the final time-step. On top of this base LSTM network, an attention layer was implemented. After computing the hidden vector at each time-step, the features are used to compute attention weights as proposed by Luong et al. [7]. The attention weights are multiplied by the hidden vectors, and a final fully connected layer converts the resulting feature vector into a singular prediction. Below is the specific function that used to compute the attention weights:

$$a_t = \text{softmax}(W_a h_t)$$

We adapted a simple genetic algorithm to update the weights of the LSTM models (Figure 5). In this algorithm, a generation of models is randomly initialized. Losses are then calculated over training data (without gradient descent). The model with the best fitness function, i.e. lowest loss over the training data, is saved and continues to the next generation. A few mutations of the best model are also added to the next generation. Finally, mutations of random choices of the best few models populate the rest of the next generation such that the generation size remains fixed. Then, the models are evaluated again and the procedure repeats. The number of networks per generation, the number of top models to consider, and the number of mutations of the best network are tunable hyperparameters. In the data we report for this study, we considered 10 networks per generation, 4 top models, and mutated the best model 4 times. To achieve mutations, we add Gaussian noise to the network parameters. The variance of this noise is also a tunable hyperparameter; in these experiments, it was fixed at $0.1$.

Though this algorithm is simple and ignores the gradient entirely, it is surprisingly effective at diminishing training loss. Due to the number of models necessary for each generation (we run the same number of generations as we do epochs for gradient descent), the GA is slower to train.

To run our experiments, we launched an AWS EC2 instance. We opted for a g4dn.xlarge instance type because of its balance of cost per hour and larger GPU.

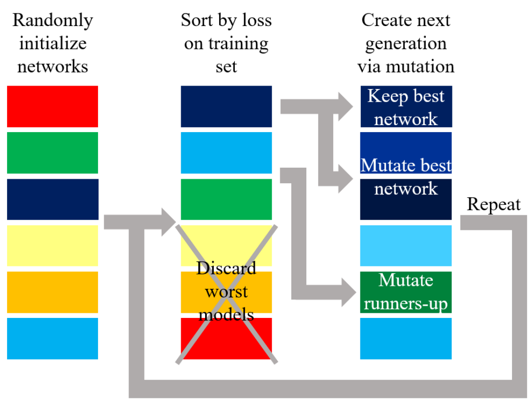

Randomly initialize networks
Sort by loss on training set
Create next generation via mutation
Keep best network
Mutate best network
Repeat
Discard worst models
Mutate runners-up

FIGURE 1. Schematic representation of genetic algorithm used to update LSTM weights.

## 4. EXPERIMENTAL RESULTS

Our results show that a base LSTM is unable to make useful forecasts of stock prices with this methodology. In Figure 2, we observe that the training loss as a function of weight updates, even when a moving average of 1000 weight updates is applied, is extremely noisy and shows little to no downward trend. The same model is able to more-or-less monotonically decrease its training loss when training on a single stock, indicating that this LSTM architecture is implemented correctly and is capable of memorizing stock histories. However, when 70+ stocks are being analyzed, this capacity for rote memorization does not generalize well to validation data. The training loss and validation loss for the base LSTM also exhibit little downward trending and are weakly correlated (Figure 2). The base LSTM correctly predicts if a stock is going up or down only 43.7% of the time, which is essentially worse than random guessing (Table 1).

The attention module improves this performance. While the training loss is still very noisy as a function of weight updates, the training loss and validation loss exhibit a clear downward trend (Figure 3). Training loss drops by 0.9% and validation loss drops by 0.3%, though validation loss remains higher than training loss as is expected. Though the attention LSTM trains better, its performance for predicting the direction of stock movement is still worse than a random coin flip: 45.8%.

The performance for the base and attention LSTMs trained by genetic algorithm exhibits similar trends to the networks' respective performances with stochastic gradient descent. For the base LSTM, the genetic algorithm leads to a training loss decrease of 0.4% and a validation loss decrease of 0.6% (Figure 2). The final validation loss is 2.7% greater than that of the final validation loss achieved with with the base LSTM. It should be noted that the average training loss of a generation of models is is not correlated to the performance of the best model, as most mutations are detrimental and not beneficial to a model's fitness function. The GA selects a base LSTM model which correctly predicts the stock movement direction 43.2% of the time —a poor performance. Models trained with GA also took significantly longer to train.

Mirroring the result for the networks trained via stochastic gradient descent, the GA trains better with the attention-based LSTM. Training and validation losses are tightly correlated, decreasing by 5% and 4% respectively. The validation loss achieved by the attention-LSTM trained via GA is 0.8% lower than the validation loss achieved by the attention-LSTM trained via SGD with the same number of ecochs as GA generations.

## 5. DISCUSSION

Our report is part of a larger ecosystem of research on time-series prediction of stock prices with cutting-edge machine learning methods. Many promising results are achieved via empirical tricks rather than deeply theoretical

| | Base | Attention | Base + GA | Attention + GA |
|---|---|---|---|---|
| Average Batch Loss | 1.180 | 0.664 | 1.236 | 1.104 |
| Directional Accuracy | 0.437 | 0.458 | 0.432 | 0.434 |

TABLE 1. Results summarized.

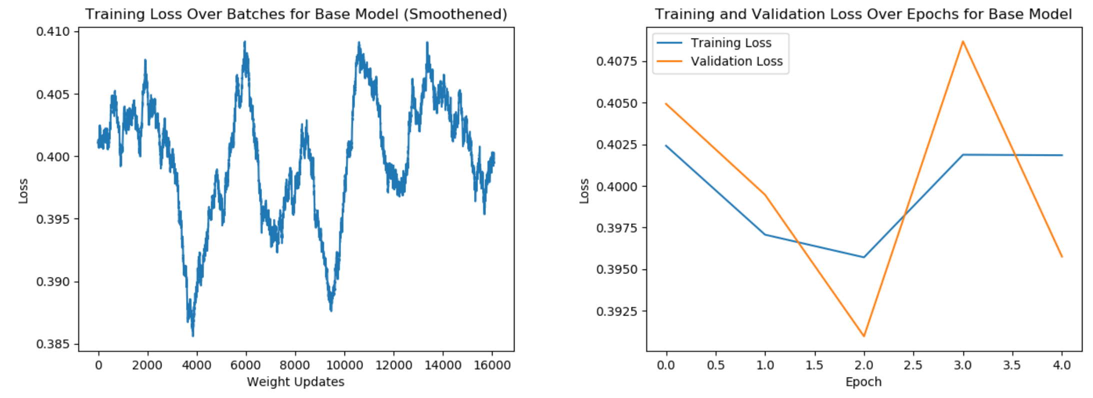


FIGURE 2. Performance on Base LSTM

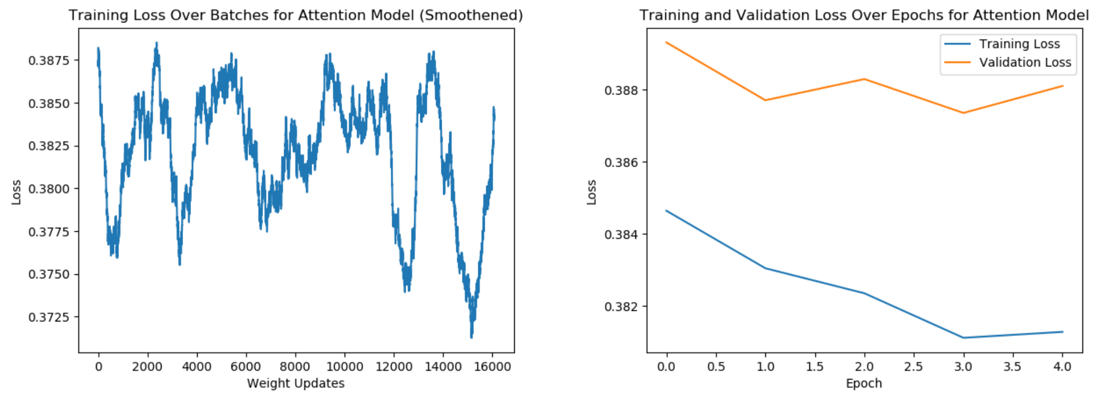
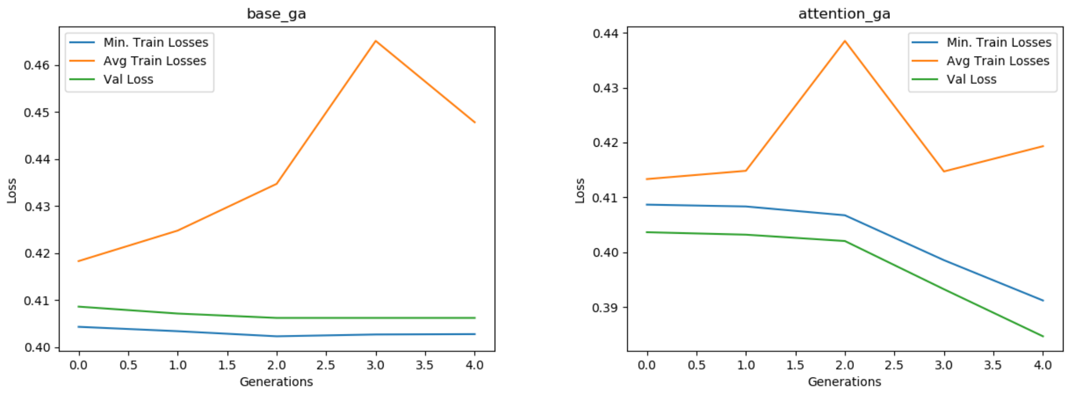


FIGURE 3. Performance on Attention LSTM

approaches; our study falls in this category. In this report we verify that attention-modules improve LSTM performance for time-series prediction; this may be because Attention layers help the network learn features from earlier time-steps that may otherwise vanish over a long time-series network. For stock market price prediction, a problem with an incredible amount of complexity, perhaps the network was able to learn correlated signals from earlier time-steps, and this may have had a marginal effect on the its loss, even though it was an ultimately uninspiring performance. We also find that genetic algorithms which ignore the gradient, even with small generation sizes, are able to decrease loss measurably in this setting. We speculate the reason for this achievement is the energy landscape of the loss given the noisy stock data. The noisy input data leads to a ”rocky” energy landscape where the performance of SGD is limited, allowing chaotic movements of a GA instantiated with several models to compete in finding minima.

Though we do not prescribe GAs for training weights of LSTMs, this finding does suggest that optimizing randomly instantiated networks in parallel or using alternate optimization algorithms such as particle swarm optimization could perform well in forecasting chaotic time-series data.

To discuss the results, all 4 models also predicted a steep drop at the beginning of the predicted sequence. This is may be because the last few days of the input was already dropping and indicated an upcoming fall in price. The stock market tends to follow a certain set of patterns, but timing it is tough. Based on the previous drops in the stock price, it



(A) Base LSTM with Genetic Algorithm (B) Attention LSTM with Genetic Algorithm

FIGURE 4. Performance on Genetic Algorithm

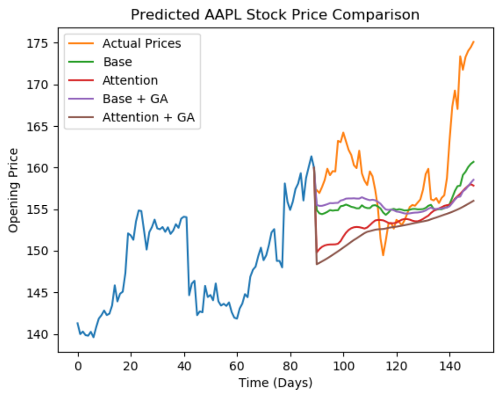


FIGURE 5. Model forecasts vs. actual future stock price (AAPL).

is understandable that the models predict a drop and then rebound. The Base + GA model dropped the least and the attention layer seemed to have the biggest effect in improving results. Furthermore, although the directional accuracies for all four models were all less than 50 percent, which is essentially worse than a randomized decision, it is important note that this accuracy measurement was taken daily. If we look at Figure 5, all 4 models estimated that AAPL's stock price would rebound and trend in a positive direction. This distinction is important for people investing for the long term, for whom daily variation does not matter.

## 6. Future Work

As discussed in the above paragraph, our model was not necessarily great at predicting next-day prices, but it did correctly predict long-term trends within a reasonable time frame. Instead of measuring day-by-day MAE loss, it may be interesting to use some measure of eventual prediction of the trend, perhaps optimizing for the accuracy of the timing. We would also like to explore training with longer sequence lengths.

This study looked specifically at training deep learning models on stocks in the Information Technology sector, which is known for being a highly volatile, growth-oriented industry. In future work, expanding this experiment to other sectors, such as Utilities or Industrials, which tend to have more stable stocks, may yield different results.

## REFERENCES

[1] K. Chen, Y. Zhou, and F. Dai, "A LSTM-based method for stock returns prediction: A case study of China stock market," in *Proceedings - 2015 IEEE International Conference on Big Data, IEEE Big Data 2015*, 2015, ISBN: 9781479999255. DOI: 10.1109/BigData.2015.7364089.

[2] D. M. Nelson, A. C. Pereira, and R. A. De Oliveira, "Stock market's price movement prediction with LSTM neural networks," in *Proceedings of the International Joint Conference on Neural Networks*, 2017, ISBN: 9781509061815. DOI: 10.1109/IJCNN.2017.7966019.

[3] Y. Qin, D. Song, H. Cheng, W. Cheng, G. Jiang, and G. W. Cottrell, "A dual-stage attention-based recurrent neural network for time series prediction," in *IJCAI International Joint Conference on Artificial Intelligence*, 2017, ISBN: 9780999241103. DOI: 10.24963/ijcai.2017/366.

[4] G. Liu and X. Wang, "A Numerical-Based Attention Method for Stock Market Prediction with Dual Information," *IEEE Access*, 2019, ISSN: 21693536. DOI: 10.1109/ACCESS.2018.2886367.

[5] K. J. Kim and I. Han, "Genetic algorithms approach to feature discretization in artificial neural networks for the prediction of stock price index," *Expert Systems with Applications*, 2000, ISSN: 09574174. DOI: 10.1016/S0957-4174(00)00027-0.

[6] H. Chung and K. S. Shin, "Genetic algorithm-optimized long short-term memory network for stock market prediction," *Sustainability (Switzerland)*, 2018, ISSN: 20711050. DOI: 10.3390/su10103765.

[7] M. T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," in *Conference Proceedings - EMNLP 2015: Conference on Empirical Methods in Natural Language Processing*, 2015, ISBN: 9781941643327. DOI: 10.18653/v1/d15-1166.