Developing Machine Learning Algorithms to Price American-Style Stock Options

ENPH 455 Fall Term Report – Engineering Physics Thesis

Student Author: Nathan Pacey

Supervising Professor: Ryan Martin

Queen’s University

Kingston, Ontario Canada

November 2023

Table of Contents

Introduction to Options and Project Motivation

This project focuses on American-style stock call and put options, which grant investors the ability to buy or sell shares at predetermined prices [1]. Predicting stock option prices, despite numerous attempts, remains challenging due to various factors, with traditional models like the Black-Scholes formula being less effective for American options due to their reliance on expiration time [2]. To enhance pricing prediction accuracy, this project utilizes machine learning algorithms, specifically recurrent neural networks (RNNs) [3]. The objective is to optimize RNNs by systematically tuning parameters and comparing various architectures to achieve precise option price predictions. These insights have the potential to benefit diverse time series data applications and contribute to the understanding of machine learning in financial markets and data analysis.

Problem Definition

This project uses many to many synched RNNs to forecast American-style stock options, focusing on 'at the money' options with short-term expirations from North American large-cap equities. It aims to predict option ask and bid prices using TensorFlow, allowing for parameter adjustments guided by iterative validation. The project also seeks to create a versatile model applicable to various equities through normalization techniques and diverse architectures. Furthermore, it compares employing a Gated Recurrent Unit (GRU) connected to a Fully Connected layer versus using a Long Short-Term Memory (LSTM) model. This comparison aims to discern the model advantages in the context of time series data analysis.

Background Information

General Machine Learning

The core component of a machine learning model is the artificial neuron, akin to a biological neuron. Neurons are organized into network layers and perform mathematical operations. Each neuron processes inputs with associated weights, applying linear transformations followed by nonlinear activation functions to generate outputs [4].

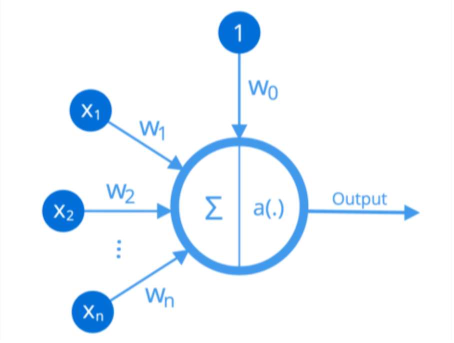


Figure 1. Single Neuron Visualization taken from Machine Learning Mastery [4].

The transformation involves multiplying each input value by its corresponding weight, summing the results, and adding a bias term [4].

$$z = \left(\sum_{i=1}^{n} x_i \times w_i \right) + b \tag{1}$$

This weighted sum, z, then undergoes a nonlinear activation function denoted as a to generate the neuron's output, a(z). The choice of activation function introduces nonlinearity into the model, enabling it to capture complex relationships in the data [4].

$$a(z) = a\left(\sum_{i=0}^{n} x_i \times w_i \right) \tag{2}$$

However, a single neuron model lacks the complexity needed for time series predictions. To achieve the necessary complexity, the model must incorporate a network of interconnected neurons and employ backpropagation, gradient descent algorithms, and input-output cycling which is encompassed in an RNN [3].

Recurrent Neural Networks and Time Series Forecasting

Recurrent Neural Networks (RNNs) are specialized artificial neural networks designed to incorporate memory and temporal dynamics for sequential data processing. They introduce directed memory cycles, unlike standard neural networks [5]. A basic RNN consists of three layers: input, hidden, and output. Sequential data enters through the input layer, undergoes processing in the hidden layer with applied activations, and can involve multiple hidden layers in complex RNN architectures. Each hidden layer operates independently with its own set of weights and biases, enabling the capture of unique data aspects [5].

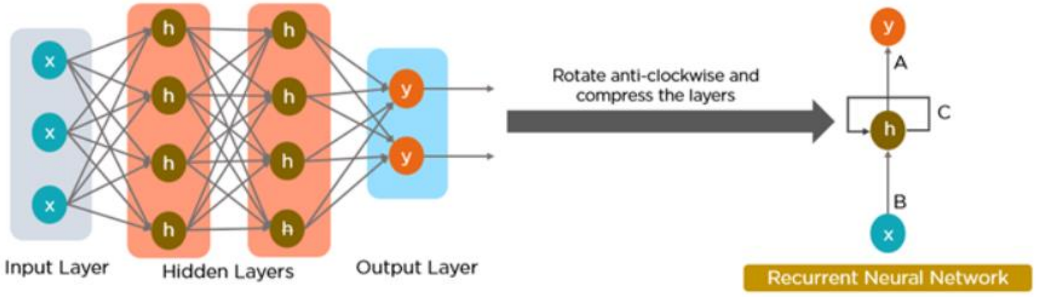


Figure 2. Visualization of a general form RNN displayed as a many to many RNN from Analytics Vidhya [3].

RNNs excel in time series forecasting by preserving temporal relationships between consecutive inputs through recurrent neurons, which store and merge past input states with the current input, capturing hidden dependencies in sequential data like stock option pricing [6].

Project Goals and Constraints

The main goal of this project is to develop a machine learning model to effectively capture the correlations or patterns between an option's parameters, historical performance, and its value to traders. The model's success could aid options traders in identifying mispriced options and arbitrage opportunities, although achieving this level of success is challenging due to market efficiency and financial institutions' interest in accurate pricing [7]. As a result, the project also aims to iterate on RNN model parameters to reduce prediction errors and contribute to broader time-series data analysis and machine learning optimization research.

Design Considerations

Although this project is centered on software, it's essential to recognize the environmental implications related to the electricity consumption required to operate ML algorithms. Additionally, the project underscores the importance of meticulous and prudent performance assessments to prevent inadvertent misinformation and financial losses, considering the potential utilization of the models by traders.

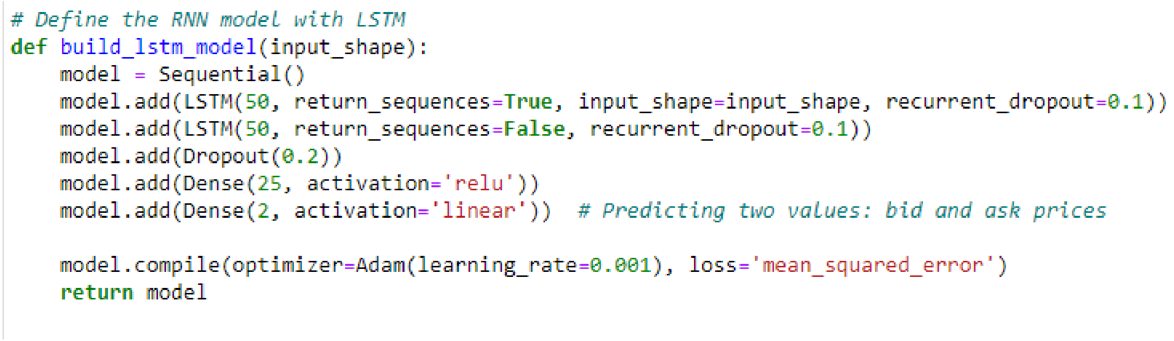
Project Progress and Design Iterations

Processing Options Data

Historical options data for 56 companies from February 2022 to December 2022 was acquired using the yfinance Python library and stored on the Neutrino server in zip files. Data processing, including the removal of out-of-the-money options, was performed using the pandas library.

Developing the LSTM model

The initial LSTM model, built with TensorFlow and Keras, closely follows Israt Jahan's stock pricing LSTM model [8]. It comprises two LSTM layers with 50 units each, along with a dropout layer to prevent overfitting. The architecture also includes two fully connected (dense) layers: one with 25 units using Rectified Linear Activation (relu) and a final layer with 2 units for predicting bid and ask prices with linear activation. The model is compiled using the Adam optimizer with a learning rate of 0.001, employing Mean Squared Error as the loss function [8].



```
# Define the RNN model with LSTM
def build_lstm_model(input_shape):
    model = Sequential()
    model.add(LSTM(50, return_sequences=True, input_shape=input_shape, recurrent_dropout=0.1))
    model.add(LSTM(50, return_sequences=False, recurrent_dropout=0.1))
    model.add(Dropout(0.2))
    model.add(Dense(25, activation='relu'))
    model.add(Dense(2, activation='linear'))  # Predicting two values: bid and ask prices

    model.compile(optimizer=Adam(learning_rate=0.001), loss='mean_squared_error')
    return model
```

Figure 3. LSTM ML model developed in python for this project.

Training and Validating the Initial Model

To train and validate the initial model, we utilized Apple (ticker AAPL) and AMD options data and conducted training over 10 epochs. Prior to evaluating the options data, it underwent a normalization process based on the minimum and maximum values within the parameter data.

The data was then split into training and testing sets using the train_test_split function from sklearn, with an 80% training and 20% testing partition, enabling model evaluation.

Following the initial training, a significant issue arose: the options data contained null values that hindered the RNN model's training process, resulting in divide-by-zero errors in the loss function. To address this, code modifications were made during the preprocessing step to exclude options with any null values in their associated parameters.

Initial Results

The results of this model were visualized using a variety of plots to understand how well the LSTM model predicted both the ask and bid prices. First, one can see the predicted versus actual bids and asks across the entire dataset along with the associated error.

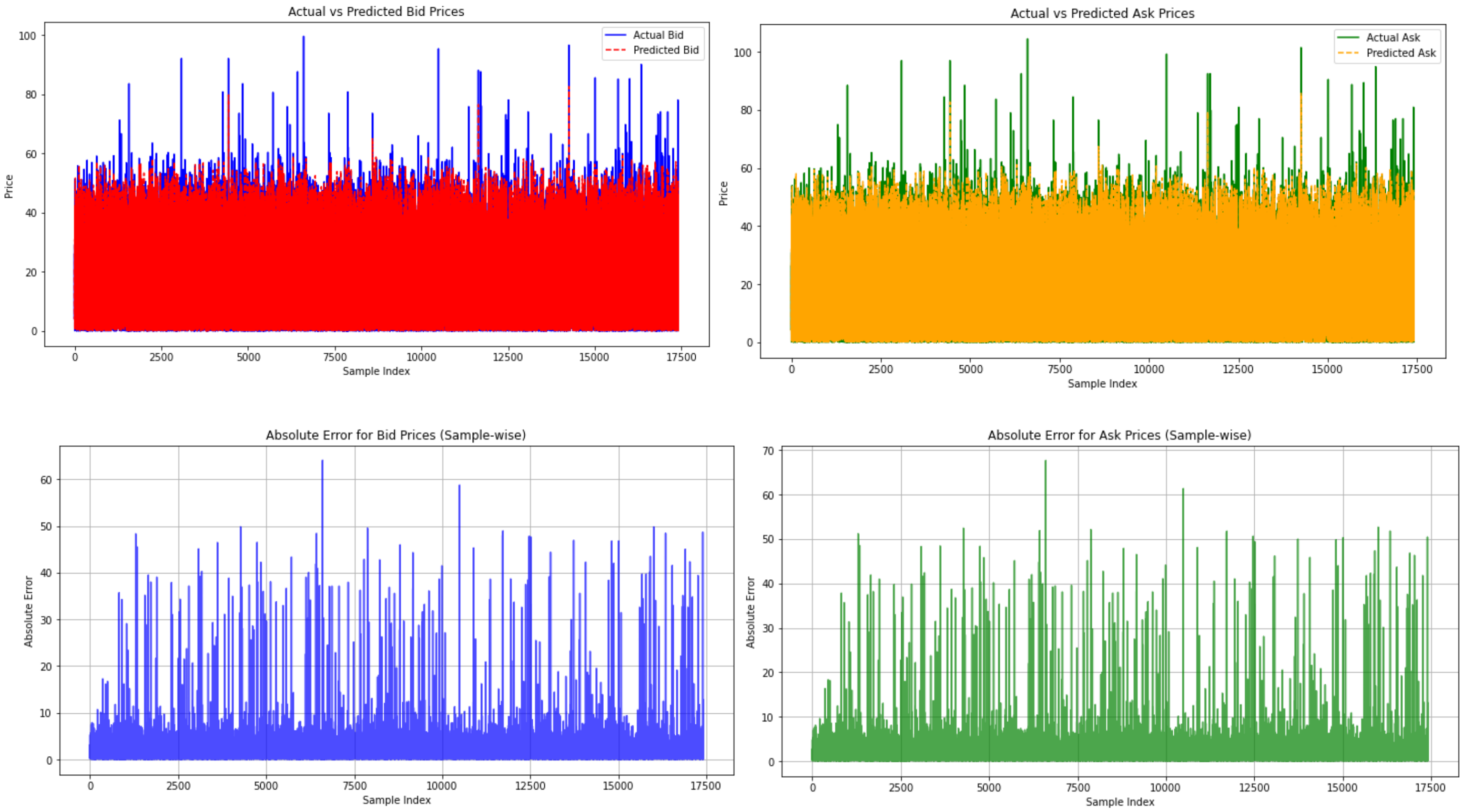


Figure 4. Initial results of the LSTM displaying the predictions against the real ask and bid values along with the associated error for each option contract.

In order to assess the model's effectiveness, both mean squared error (MSE) and mean absolute error (MAE) metrics were computed using the sklearn.metrics function. Where a lower mean squared error indicates more accurate model predictions. The initial model resulted in a MSE of 21.04 and a MAE of 2.30.

While significant deviations from the actual bid prices are evident when analyzing a larger dataset, a closer examination of the first 100 samples reveals that the model effectively tracks the real bid and ask prices with accuracy.

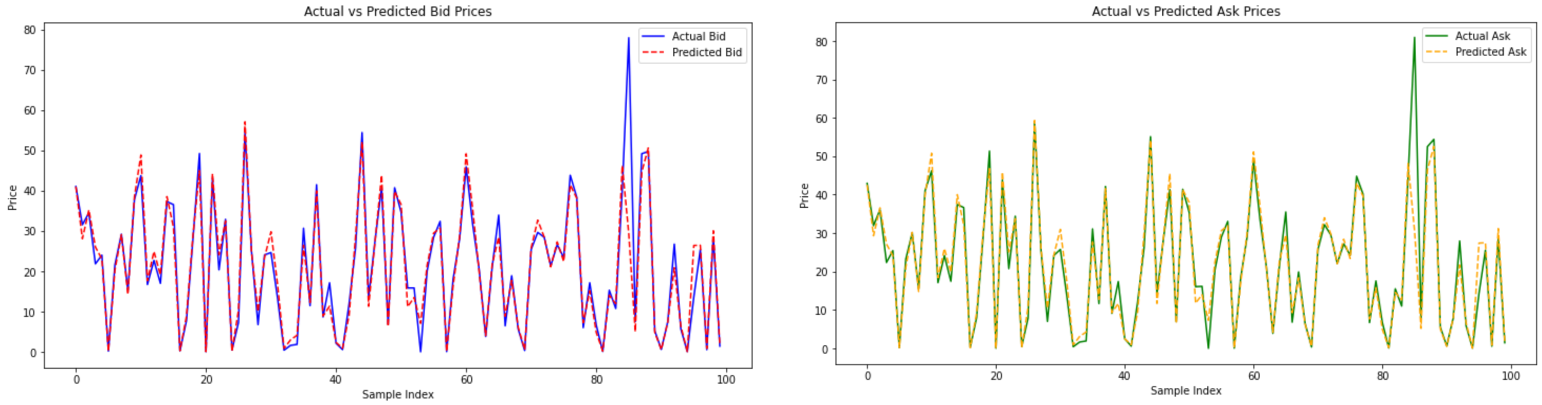


Figure 6. Real ask and bid values plotting along with the initial models predicted values for the first 100 option contracts in the dataset.

To gain deeper insights into the distribution of ask/bid predictions and their corresponding errors across the dataset, histogram plots were generated.

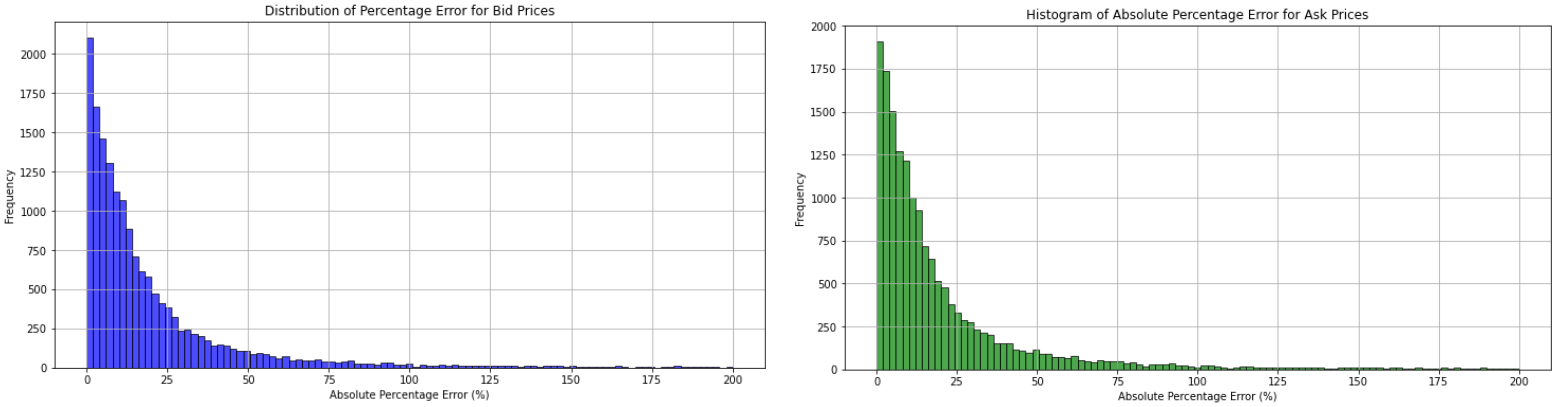


Figure 5. Histogram plots of the frequency of percentage error distribution of the model's bid and ask predictions.

The majority of errors, both for bid and ask predictions, fall within a 20% error range, indicating the reasonableness of the RNN model choices. However, it's important to note that these results, while promising, may not fully reflect real trading conditions. To better simulate a real scenario, the testing data should not be randomly selected from the dataset.

Model Refinement and Testing on Realistic Data

To emulate real-world conditions, the testing data was selected as the final 20% of the entire dataset, subjecting it to the same training process. This approach yielded expectedly less favorable results.

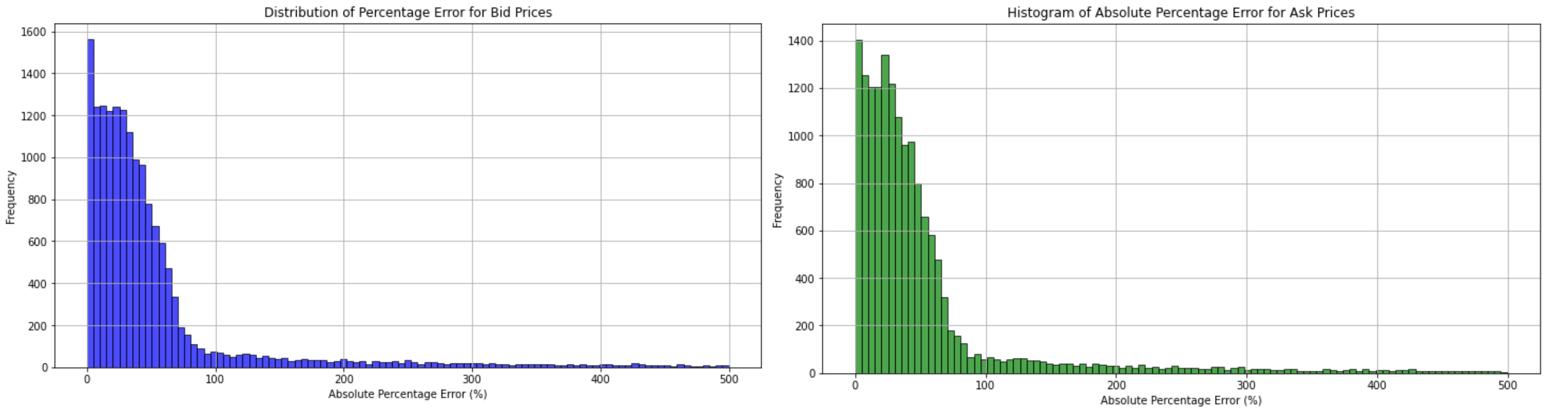


Figure 7. Distribution of percentage error of bid and ask predictions for the model tested with the more realistic training testing data split.

To assess the efficacy of these models, evaluations were conducted on a completely different dataset, specifically Amazon options. Amazon was selected as the test dataset due to its comparable trading volume, volatility, and industry characteristics to AAPL and AMD.

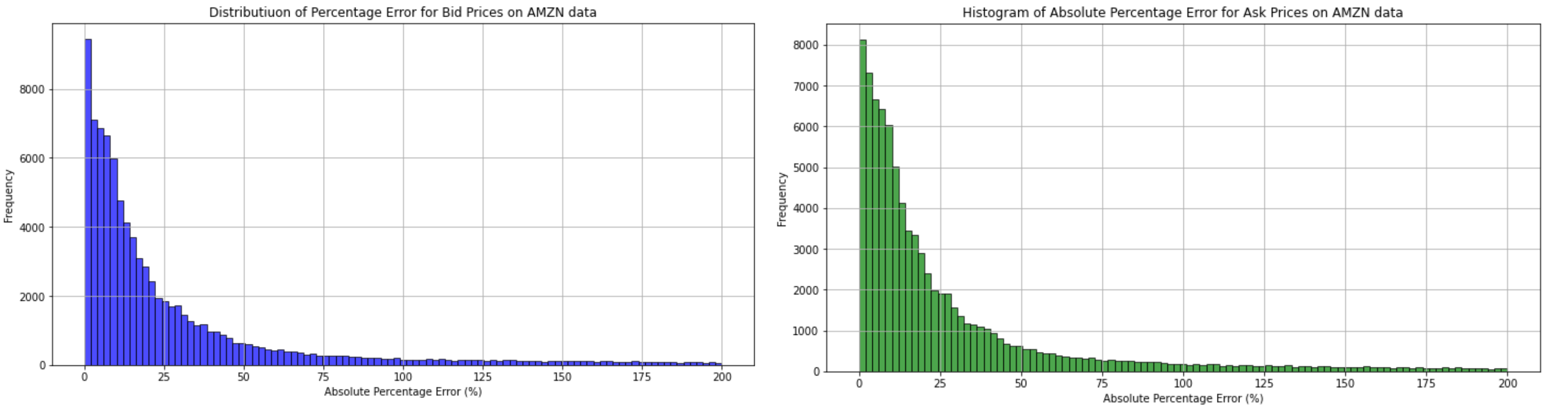


Figure 8. Distribution of percent error of bid and ask predictions for Amazon stock option predictions.

These findings demonstrate the model's ability to extract patterns from previously unobserved stock options data. Upon closer examination of the initial 100 samples from the Amazon dataset, the LSTM model exhibits a reasonable level of accuracy in predicting the ask/bid spread dynamics over time.

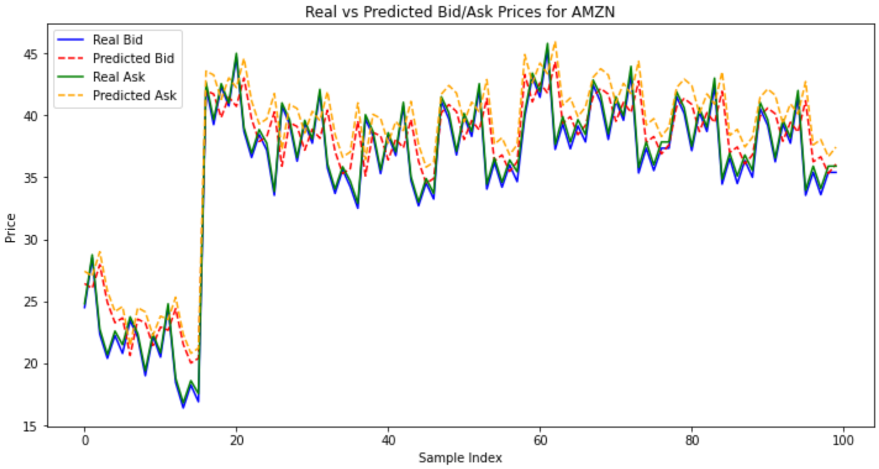


Figure 9. Real bid/ask data for Amazon options plotted along with the models' predictions for the first 100 contracts.

Specifically, the MSE was about 24.14 and MAE was about 3.02 which is comparable to the randomized testing on Apple and AMD data. For the full code including model iterations and results see the GitHub repository[1].

Future Progress

LSTM Optimization

The higher error observed in the realistic Apple and AMD data split compared to Amazon may seem counterintuitive but can be attributed to various factors. One reason is the diversity of option types and parameters in the last 20% of data, differing significantly from the first 80%, posing pricing challenges for the model. Market movements of large-volume tech stocks like AAPL and AMD often exhibit similar percentage changes during the same period, affecting the model's performance.

This raises questions about optimal data splitting. One approach involves using the newest 20% of data for each ticker to better simulate real-life scenarios. To improve prediction models, a larger training dataset is needed to accommodate various option parameter weightings and reduce errors in realistic data splits.

Stock options are sensitive to stock price nuances like volatility and volume, making linear parameter scaling inadequate. Further research is needed to integrate multiple ticker data into a single model, possibly through normalization functions or diverse training on stock prices and parameters.

Developing a Hybrid GRU and Fully Connected Model

An alternative approach will be developed alongside the LSTM model. This approach isolates non-temporal parameters, such as date, time, contract name, time until expiration, and ticker, from the time-

[1] Code repository for the project: github.com/NathanPaceydev/ML-for-American-Style-Stock-Options

varying ones. The model processes time-dependent parameters using a functional GRU RNN and then feeds the output into a fully connected layer.

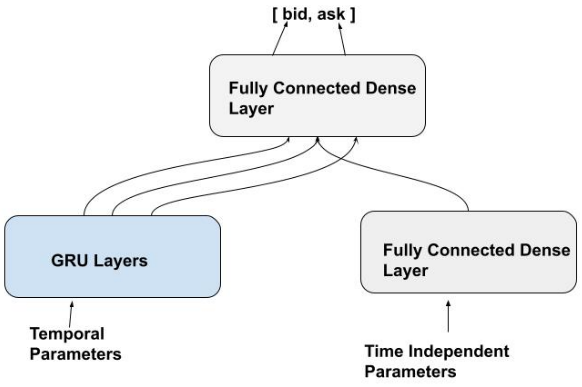

[bid, ask]
Fully Connected Dense Layer
GRU Layers
Fully Connected Dense Layer
Temporal Parameters
Time Independent Parameters

Figure 10. Hybrid Functional RNN model splitting time dependent parameters into a GRU and passing output along with static parameters to a Fully Connected Layer.

This model's advantage lies in effectively segregating the daily market movements by distinguishing temporal parameters. This separation can potentially reduce errors in scenarios involving testing data from the latest contract dates. Comparing the performance of this model to the LSTM model could provide valuable insights into strategies for predicting real-time dependent data. A revised project plan for the remaining duration of the project can be found in Appendix A.

Conclusion

In summary, this project has focused on the development and testing of machine learning models, particularly recurrent neural networks (RNNs), for predicting American-style stock options' ask and bid prices. The project has made significant progress in building an initial LSTM model and exploring various aspects of data preprocessing, model architecture, and evaluation.

Moving forward, the project will continue to refine and optimize the existing models, address challenges such as diverse data and model performance, and explore alternative approaches, including the development of a hybrid GRU and Fully Connected model. The project's journey of iterative design and testing remains ongoing as it strives to achieve more accurate and reliable predictions for American-style stock options.

References

[1] J. Chen, "What are Options? Types, Spreads, Example, and Risk Metrics," Investopedia, 24 April 2023. [Online]. Available: https://www.investopedia.com/terms/o/option.asp. [Accessed 13 October 2023].

[2] A. Hayes, "Black-Scholes Model: What It Is, How It Works, Options Formula," Adam Hayes, 5 May 2023. [Online]. Available: https://www.investopedia.com/terms/b/blackscholes.asp. [Accessed 12 October 2023].

[3] D. Kalita, "A Brief Overview of Recurrent Neural Networks (RNN)," Analytics Vidhya, 3 August 2023. [Online]. Available: https://www.analyticsvidhya.com/blog/2022/03/a-brief-overview-of-recurrent-neural-networks-rnn/. [Accessed 13 October 2023].

[4] S. Cristina, "Calculus in Action: Neural Networks," Machine Learning Mastery, 16 March 2022. [Online]. Available: https://machinelearningmastery.com/calculus-in-action-neural-networks/. [Accessed 22 November 2023].

[5] V. Madisson, "Stock market predictions with RNN using daily market variables," Towards Data Science, 11 June 2019. [Online]. Available: https://towardsdatascience.com/stock-market-predictions-with-rnn-using-daily-market-variables-6f928c867fd2. [Accessed 11 October 2023].

[6] M. Saeed, "An Introduction to Recurrent Neural Networks and the Math That Powers Them," Machine Learning Mastery, 6 January 2023. [Online]. Available: https://machinelearningmastery.com/an-introduction-to-recurrent-neural-networks-and-the-math-that-powers-them/. [Accessed 13 October 2023].

[7] L. Downey, "Efficient Market Hypothesis (EMH): Definition and Critique," Investopedia, 24 April 2023. [Online]. Available: https://www.investopedia.com/terms/e/efficientmarkethypothesis.asp. [Accessed 18 November 2023].

[8] I. Jahan, "Stock Price Prediction Using Recurrent Neural Networks," North Dakota State University, Fargo, 2018.

[9] M. Lesuisse, "What are the advantages of pricing American options using artificial neural networks?," Université de Liège, Liège, 2022.

[10] Tensorflow, "Time series forecasting," Tensorflow, 27 July 2023. [Online]. Available: https://www.tensorflow.org/tutorials/structured_data/time_series. [Accessed 9 October 2023].

[11] Quant Next, "Artificial Neural Network for Option Pricing with Python Code," Youtube, July 2023. [Online]. Available: https://www.youtube.com/watch?v=r6KTnKim_BA&ab_channel=QuantNext. [Accessed 3 October 2023].

Appendix A.

Main Body Word Count Without Captions

1778 words

Updated Project Plan

Remaining Project Duration: ~ 4 Months

Month 1: Data Cleanup and LSTM Optimization

1. Week 1-2: Data Cleaning and Preparation

   - Address null or garbage values in the entire options data.

   - Modify preprocessing code to handle any data issues.

2. Week 3-4: Initial LSTM Model Optimization

   - Optimize the LSTM model based on insights from preliminary training.

   - Train and Test LSTM Model using entire dataset.

   - Explore different hyperparameters and architecture variations.

Month 2: Model Testing and Normalization Research

3. Week 1-2: Testing and Validation of LSTM

   - Assess the optimized LSTM model's performance using various metrics.

   - Validate results on different datasets.

   - Compare with previous findings.

4. Week 3-4: Research on Normalization Techniques

   - Investigate and research various normalization techniques for options data.

   - Evaluate the applicability and potential benefits of different normalization methods.

Month 3: Hybrid Model Development and Refinement

5. Week 1-2: Develop a Hybrid GRU and FC Model

- Design the architecture for the hybrid model.

- Implement GRU and Fully Connected layers.

- Prepare to integrate it into the project for testing.

6. Week 3-4: Model Refinement and Validation

- Fine-tune both LSTM and Hybrid models based on research and insights.

- Validate the hybrid model's performance on relevant datasets.

- Address any challenges or issues encountered during development.

Month 4: Model Optimization and Comparison

7. Week 1-2: Further Model Optimization

- Optimize both LSTM and Hybrid models based on research findings.

- Fine-tune hyperparameters and architecture elements.

8. Week 3-4: Model Comparison and Conclusion

- Compare the performance of the LSTM and Hybrid models.

- Assess their suitability for pricing American-style stock options.

- Summarize key findings and achievements in the final report.

Throughout the Project: Continuous Documentation and Iteration

- Maintain detailed documentation of all code, changes, and insights.