Pricing of European Options using a Generative Adversarial Model

A Research Proposal

Submitted as part of the requirements for:

CE902 Professional Practise and Methodology

Name: Alexandros Antoniou

Supervisor: Dr Maria Kyropoulou

Date: 20th March 2020

Abstract

This paper proposes the employment of generative adversarial networks in the pricing of vanilla European options. Options represent a large part of the stock exchange and for that reason accurately and efficiently determining their prices has become an increasingly important task. With the advent of Machine Learning and subsequently Deep Learning incredible success has been achieved in the pricing of stocks and predicting the behaviour of the stock market. In this research proposal, we propose a generative adversarial architecture with Deep Convolutional Generative Adversarial Network (DCGAN) as the generator and the discriminator for the pricing of options. The generator is fed data on past, existing options from a selected European stock market and generates realistic data belonging to the same lognormal distributions, while the discriminator categorises them as generated or real data. We collect data on a handful of FTSE-250 options available over a sufficiently wide range of trading days attempting to estimate the price of the options in the future. Efficiency and accuracy metrics are obtained by comparing our results with more traditional methods of options pricing, mainly to the Black-Scholes model.

Keywords: Deep Learning; Generative Adversarial Networks; Options pricing; Data Mining; Neural Networks

Contents

1 Introduction

Options are one of the many financial instruments constantly involved in trades inside stock markets all around the world. As such, investors have spent an incredible amount of time and thought into designing models that would accurately and efficiently predict the behaviour of these instruments and how they relate to their underlying assets.

This paper is divided into 4 main sections. Firstly, a review of existing literature on the topic of options pricing and generative adversarial networks. The literature review provides the reader with adequate resources to help them digest the contents of the remaining paper. Work referenced in this section provide in-depth explanations for any technical detail the reader encounters in later sections. The review will also look back into the history and evolution of the techniques used in this paper, such as the history of neural networks as they relate to generative adversarial models, as well as the traditional methods for pricing options, namely the Black-Scholes-Merton model and the differential equations that arise from their assumptions.

The second section provides the reader with the goals and objectives of the research. The goals are derived from related works referenced in the previous section and the questions the authors posed in their own research, possibly attempting to enhance current literature. Objectives are then set based on these goals and research questions posed in this paper are to be resolved by the end of this project. Limitations of any model used are also to be mentioned in this section.

The third section contains the methodology used to undertake this project. This includes any data used in the research and how they were selected, how the types of networks to be employed were selected and a detailed explanation of how the networks operate and produce the desired output. We also explain the selection of tools to be used, such as the language the networks are coded in and any framework or software used.

The fourth section describes the evaluation process for the models. This section provides descriptions of the performance metrics selected to evaluate the efficiency and accuracy of the fully trained generative model. More traditional means of options pricing will be utilised to compare against our own model. Any and all testing of the model will be catalogued, such as unit testing of both networks, generator and discriminator.

Finally, the fifth section provides a plan for the development and deployment of the project. The developmental process will be broken down into a series of smaller sub-goals to be completed during the Summer term of the academic year. We provide a comprehensive list of tasks to be delivered, showing the work breakdown for this project, including a Gantt chart at the end of the paper.

2 Literature Review

A. Traditional Means of Options Pricing

In his seminal 1900 PhD thesis, Louis Bachelier introduced a novel model for the valuing of stock options using Brownian motion, a stochastic process. This instance is historically the first paper to introduce advanced mathematics in the context of finance, making Bachelier the forefather of mathematical finance. In that paper, Bachelier employed random walks to model stock prices in discrete time, in the process showing that random walks can generate Brownian motion. One deeply important assumption Bachelier made was that stock prices follow a *normal distribution* which can yield realistic predictions for prices in short time periods. However, this model cannot cope with the fact that stock prices cannot have negative values, failing its predictions in longer time periods. Bachelier used the example of the *simple* option, where the option's premium equals the option's spread and concluded that the prices of options must be proportional to the square root of the time the option has been in trading[1].

It is this stochastic modelling of options and stocks that gave rise to the Efficient Market Hypothesis which lies at the centre of modern financial economics. The Efficient Market Hypothesis states that asset prices reflect all available information in the market, implying that predicting stock prices on a risk-neutral basis is hard, or even impossible. Mandelbrot (1963) and Samuelson (1965)[2] both provide further insights into this seemingly fundamental, yet unstestable hypothesis at the core of financial economics. Mandelbrot used *lognormal distributions* as the basis of his model for pricing commodities and completely avoided the problem of negative price values, encountered by Bachelier[3]. Eugene Fama later formalised the Efficient Markets model with his 1970 paper reviewing earlier theory and empirical work. Fama finds that price changes in stocks appear to be more random than previously thought, as they are discovered to be best modelled through random walks. His paper concludes that little evidence exists against the hypothesis[4].

Note the 60-odd year difference between Bachelier's work and Mandelbrot, Samuelson and Fama. The valuation of options became a tricky science as, to value a derivative instrument, one needs to know how the underlying asset behaves. Many came close to a solution, although they were found to be incomplete.

Breakthrough was finally made with Fischer Black and Myron Scholes in their ground-breaking paper titled *The Pricing of Options and Corporate Liabilities*[5], which was later expanded on by Robert Merton who coined the term "Black-Scholes options pricing model." Building on earlier work, Black and Scholes base their work on a few assumptions about the stock market. Firstly, they assume that the option in question is a European call option. Call options are contracts that give the holder the right to purchase an amount of the underlying asset at a set date, called the maturity date[6]. An important assumption they make is that the price of the option is a function of a stock whose price follows a random walk, continuous in time and with variance rate proportional to the square of the underlying stock's price. This allows for a *log-normal distribution* on the possible stock prices which bypasses the negative prices problem Bachelier had and makes the variance rate of the return on the stock constant. Black and Scholes also assume there is no friction in the transactions, there is no fee to pay for exercising the option or letting it expire.

Using these assumptions, Black and Scholes construct a risk-neutral portfolio using a hedged position of one stock in the long position and:

$$1/f_1(S,t)$$

Where $f_1(S,t)$ is the first order partial derivative of the option pricing function, with regards to S, the current price of the underlying stock. As S and t change over time, the hedged position needs to be readjusted, to eliminate any risk introduced into the portfolio. Using stochastic calculus, the authors expand the portfolio's equity calculation to the now famous Black-Scholes partial differential equation:

$$\frac{\partial f}{\partial t} + rS\frac{\partial f}{\partial S} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 f}{\partial S^2} = rf$$

Where f is the option pricing function, S is the current price of the underlying stock, r is the risk-free rate and t is the time (Black & Scholes, 1973). Solving this partial differential equation yields the price of a call option, as a function of stock price and time:

$$f(S,t) = SN(d_1) - Ke^{t(T-t)}N(d_2)$$

Where N is the cumulative normal density function, and:

$$d_1 = \frac{ln\left(\frac{S}{K}\right) + \left(r + \frac{1}{2}\sigma^2\right)(T-t)}{\sigma\sqrt{T-t}}$$

$$d_2 = d_1 - \sigma\sqrt{T-t}$$

A Nobel Prize in Economic Sciences was awarded for their contributions in 1997[7].

B. The Advent of Generative Adversarial Networks

The first appearance of a generative adversarial architecture was in Ian Goodfellow's 2014 PhD thesis *Generative Adversarial Nets*. Generative Adversarial Networks arose as an attempt to create generative models for unsupervised learning, with Goodfellow *et al* presenting a novel framework in which two networks are trained simultaneously:

- A generative network *G* which tries to generate a distribution of data, and
- A discriminative network *D* which tries to assess whether a sample data belongs to the generated distribution or the training data

The two networks are then locked into a minimax game, where *G* tries to maximize the probability that *D* makes a mistake and *D* tries to minimise the probability that a sample data was generated by *G*[8]. Goodfellow gives the analogy of the generative model being a "team of counterfeiters, trying to print fake currency" and the discriminative model as "the police, trying to detect the counterfeits."

Deep Convolutional Generative Adversarial Networks are a subclass of GANs comprised entirely of CNNs. First described by Radford *et al.* in *Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks*, the architecture consists of strided convolution layers with LeakyReLU activation functions for the discriminator and convolutional-transpose layers with ReLU activation functions for the generator[9]. DCGANs

have seen most of their usage in image generation including portraits, advertisement scenes and even medical imagery[10, 11] and audio synthesis[12]. Fundamentally, GANs deal with data distributions, so any time series can be passed as input to the networks. In a financial context, we find numerous attempts in predicting stock prices, pricing of miscellaneous financial instruments like options[13, 14, 15, 16, 17].

3 Goals and Objectives

A. Overall Goal of the Project and Research Questions

The goal of the project is to develop and train a Generative Adversarial Network that accurately and efficiently prices European call and put options compared to other methods for options pricing, namely the Black-Scholes Model.

Ultimately, this isn't an attempt to create new knowledge in a financial or deep learning context, but more of an implementation of Deep Convolutional Generative Adversarial Networks to pricing simple European-style options.

American options differ from European options in the sense that they can be exercised early; European options can only be exercised at maturity. The Black-Scholes model wasn't designed to handle this extra feature, so American options will not be addressed in this paper.

B. Breakdown of Objectives

i. Build a web scraper

The first step to complete is to build a simple web scraper to collect data programmatically. The web scraper should make requests to websites holding such data on options and using APIs or manually, collect specific data points for use later in the project.

ii. Develop a simple DCGAN template

Starting with the discriminator, we build the GAN network starting from simple premade networks and layers. Testing and development are done simultaneously to make code writing more efficient.

iii. Train and fine-tune model hyperparameters

We train the networks on the data we've collected through our web scraper and fine-tune the model's hyperparameters in an attempt to optimise our network's training time and accuracy.

iv. Create logging module to collect statistics on the efficiency of the networks

We log system statistics for later evaluation and monitoring. Most programming languages offer packages in their standard library tailored to the creation of logs.

v. Implement Black-Scholes model for evaluation purposes

We evaluate our own network via comparison to an already established, traditional method for pricing options, the Black-Scholes Model. We provide the model with the same testing data we fed our GAN.

vi. Implement error metrics for the models

Later in this paper we mention error metrics for evaluating the accuracy and efficiency of the two models used, our GAN and Black-Scholes.

vii. Create a report showcasing our findings

A final report is part of the CE901 module and acts as the dissertation for this degree. The report is expected to present the complete methodology for the project, as well as go in even more detail regarding related works.

4 Methodology

This section will introduce any technologies used and methodologies adopted for the development of the project. Overviews will be provided for the programming language chosen, any external libraries used, and the development principles and practices used.

This section also contains a high-level overview of the project, including how the data is collected and a brief description of the networks to be developed.

A. Software

i. Programming Language

Python is the language of choice for the creation, training and testing of the two networks. Python is an interpreted, high-level programming language that has been gaining popularity, especially with the machine learning industry. Python is currently at its 3rd major release and although version 3.8.2 is released[18], the version used in the project is Python 3.7.4.

ii. PyTorch

PyTorch is a popular open-source Machine Learning framework for Python developed by Facebook's AI Research lab, providing Tensor Computation with both GPU and CPU support. PyTorch boasts a comprehensive Deep Learning platform that combines flexibility and speed. PyTorch utilises 3 main modules: *Autograd*, which uses automatic differentiation to calculate gradients for neural networks, *Optim*, which holds optimization algorithms, and *nn*, which makes designing neural networks and computing gradients easier and simpler than just working with the lower-level *Autograd* module[19]. This project uses PyTorch version 1.4.0.

iii. Numpy and Pandas

Numpy is a third-party library for the Python programming language, described as the quintessential scientific computing package for Python. It has strong computational capabilities, including linear algebra, Fourier transforms and random number generation[20]. The project will use NumPy version 1.18.2.

Pandas is an open-source data analysis suite for the Python programming language. Pandas provides a fast and efficient method for reading and manipulating data, along with tools to write to various data types. Its DataFrame object lies at the core of the package, optimised for performance using Cython and C[21]. The project will use Pandas version 1.0.3.

iv. Google Colaboratory

Google's *Colaboratory* is an online Jupyter Notebook environment hosted by Google that allows users to develop powerful Python code, utilising Google's extremely powerful computing resources. Google *Colab* supports several popular packages, like NumPy, Pandas, PyTorch, TensorFlow and Keras, all used in Machine Learning projects[22].

B. Development Practices and Principles

Agile Software Development is a set of frameworks based on the *Manifesto for Agile Software Development*, written in 2002. Agile emphasizes the importance of the self-organizing cross-functional team, how its members collaborate, and the customer. The Agile manifesto includes the 12 principles that shape agile software development which promote certain qualities in code writing and general development. The principles highlight the customer communication aspect of developing a product, as well as periodic meetings with the whole team to keep everyone up to date on the status of the project[23, 24].

C. Overview of the Project

i. Acquiring data

The discriminator network is trained on European-style options. We collect historical and current data on available options through Yahoo Finance. From the data, we collect specific variables to be used as the input for the discriminator and in the Black-Scholes model for evaluation, including STRIKE_PRICE, DAYS, CLOSE_PRICE, VOLUME and INTEREST.

ii. Discriminator

The discriminator *D* is a classification network with a single output in the form of a scalar. This output represents the probability that the data are generated or real. *D* takes an array of 5 inputs. The inputs are then processed through a series of Convolution layers, Batch Normalisation and Rectified Linear Units (ReLU). Following Radford *et al.*, we use strided convolution instead of standard max pooling to downsample. Finally, the network outputs the scalar probability through a Sigmoid activation function.

iii. Generator

The generator *G* samples a standard normal distribution called a latent space vector and attempts to map it to data space. *G*'s goal is to generate samples from an estimated distribution, to fool the discriminator network.

Goodfellow *et al* (2014) describe the minimax game *D* and *G* play as *D* maximising the probability that it correctly differentiates between generated samples and real samples, while *G* minimises the probability that its generated samples pass through the discriminator's tests.

From their paper, the GAN loss function is:

$$minmax\, V(D,G) = \boldsymbol{E}_{x\sim p_{data}(x)}[logD(x)] + \boldsymbol{E}_{z\sim p_z(z)}\left[log\left(1 - D\big(G(z)\big)\right)\right]$$

The solution to this game is reached when $p_g = p_{data}$, and the probability the discriminator network correctly guesses is $\frac{1}{2}$, meaning that at this point, the discriminator randomly guesses.

5 Evaluation

This section discusses the various forms of evaluation the networks will go through.

A. Integration of components

The project is split into 3 major components and the testing of the interactions between these components needs to be thorough and clean. These three components perform the following functions:

- Fetch data from stock markets
- Clean up data and filling up empty values
- Feed the clean data into the GAN

Testing must take place at each step, to ensure the minimisation of errors and losses due to factors external to the networks.

B. Unit Testing

Evaluation and development will be carried out simultaneously as per *Test-Driven Development* practices. These practices encourage the writing of small and simple *units* of code that are easy to debug and refactor when necessary. Since the project deals with stochastic methods, unit testing might not be applicable to every single part of the code but should be maximised in its use.

The built-in *unittest* Python testing suite will be used to perform unit tests.

C. Validation Testing

Finally, the generative and adversarial networks must be evaluated. The generated options prices are to be compared to more traditional pricing methods like the Black-Scholes model and errors computed, showing the difference between the two models.

The evaluation metric used will be the Mean Absolute Error and the Root Mean Square Error.

$$MAE = \frac{1}{N}\sum_{i=1}^{N} |\hat{y} - y_i|$$

$$RMSE = \sqrt{\frac{1}{T}\sum_{t=1}^{T}\left(\frac{\hat{Y}_{t+1} - Y_{t+1}}{Y_{t+1}}\right)^2}$$

Both Black-Scholes and our GAN will be compared to real prices of the options used in the project. The lower the error metrics, the closer the generated values are to the real values.

The project is also thoroughly checked to make sure that all goals and objectives set by this paper are cleared and delivered by the established due date.

6 Project Plan

The developmental phase of the project begins with the start of the summer term. Prototypes of the two networks are expected to be delivered by the start of the summer term. Due to the recent pandemic and the postponing of the summer exams however, the schedule for this project may be altered at a future date.

Development of the networks is to be completed in the months of June and July, with testing and further evaluation to be performed in August.

The deadline for this project is on August 25, 2020.

Please refer to the appendix for the Gantt chart and the Work Breakdown Structure detailing the developmental phase of the project, listing all tasks to be delivered.

7 References

1 Bachelier L, (1900). "The Theory of Speculation", *Annales scientifiques de l'Ecole Normale Superieure*, Vol. 3, issue 17, pp: 21-86.

2 Samuelson P, (1965), "Proof that Properly Anticipated Prices Fluctuate Randomly", *The World Scientific Handbook of Futures Markets*, Vol. 5, pp. 25–38. Available at: https://doi.org/10.1142/9789814566926_0002

3 Mandelbrot B, (1963). "The Variation of Certain Speculative Prices", *The Journal of Business*. Vol. 36, issue 4, pp: 394-419. Available at: https://doi.org/10.1086/294632.

4 Fama E, (1970). "Efficient Capital Markets: A Review of Theory and Empirical Work", *Journal of Finance*, Vol. 25, issue 2, pp: 383-417. Available at: https://doi.org/10.2307/2325486

5 Black F, Scholes M, (1973). "The Pricing of Options and Corporate Liabilities", *Journal of Political Economy*, Vol. 81, issue 3, pp: 637-654. Available at: https://doi.org/10.1086/260062

6 The Sveriges Riksbank Prize in Economic Sciences in Memory of Alfred Nobel 1997. (1997). Retrieved March 13, 2020, Available at: https://www.nobelprize.org/prizes/economic-sciences/1997/press-release/

7 Kuepper J, (2020). "Call Option Definition". Accessed: 16th March 2020, Retrieved from: https://www.investopedia.com/terms/c/calloption.asp

8 Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y, (2014). "Generative Adversarial Networks", *Proceedings of the International Conference on Neural Information Processing Systems (NIPS 2014)*, pp: 2672-2680. Available at: https://papers.nips.cc/paper/5423-generative-adversarial-nets

9 Radford A, Metz L, Chintala S, (2015). "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks", *Proceedings of the International Conference on Learning Representations (ICLR 2016)*. Available at: https://arxiv.org/pdf/1511.06434.pdf

10 Salimans T, Goodfellow I, Zaremba W, Cheung V, Radford A, Chen X, (2016), "Improved techniques for training gans". *Advances in neural information processing systems*, pp: 2234-2242. Available at: http://papers.nips.cc/paper/6125-improved-techniques-for-training-gans.pdf

11 Yi X, Walia E, Babyn P, (2019). "Generative adversarial network in medical imaging: A review", *Medical image analysis.* Available at: https://arxiv.org/abs/1809.07294

12 Engel J, Agrawal K K, Chen S, Gulrajani I, Donahue C, Roberts A, (2019). "Gansynth: Adversarial neural audio synthesis". Available at: https://arxiv.org/abs/1902.08710

13 Koshiyama A, Firoozye N, Treleaven P, (2019). "Generative Adversarial Networks for Financial Trading Strategies Fine-Tuning and Combination". Available at: https://arxiv.org/abs/1901.01751

14 Li J, Wang X, Lin Y, Sinha A, Wellman M P, (2018). "Generating Realistic Stock Market Order Streams". Available at: https://openreview.net/forum?id=rke41hC5Km

15 Marti G, (2019). "CorrGAN: Sampling Realistic Financial Correlation Matrices Using Generative Adversarial Networks". Available at: https://arxiv.org/abs/1910.09504

16 Takahashi, S., Chen, Y., & Tanaka-Ishii, K. (2019). "Modeling financial time-series with generative adversarial networks", *Physica A: Statistical Mechanics and its Applications*, Vol. 527 Available at: https://doi.org/10.1016/j.physa.2019.121261

17 Wiese M, Knobloch R, Korn R, Kretschmer P, (2019). "Quant GANs: deep generation of financial time series". Available at: https://arxiv.org/abs/1907.06673

18 Python Documentation (2020), Available at: https://docs.python.org/3/

19 PyTorch Documentation (2020), Available at: https://pytorch.org/

20 Numpy Documentation (2020), Available at: https://numpy.org/

21 Pandas Documentation (2020), Available at: https://pandas.pydata.org

22 Google Colaboratory FAQ (2020), Available at: https://research.google.com/colaboratory/faq.html

23 Beck K, (2002). Test-Driven Development by Example, *The Addison-Wesley Series*. Boston

24 Agile Alliance, (2020). “Agile 101”, *Agile Alliance: Agile Essentials*. Available at: https://www.agilealliance.org/agile101/

25 Manifesto for Agile Software Development, (2020). Available at: http://agilemanifesto.org/

8 Appendix

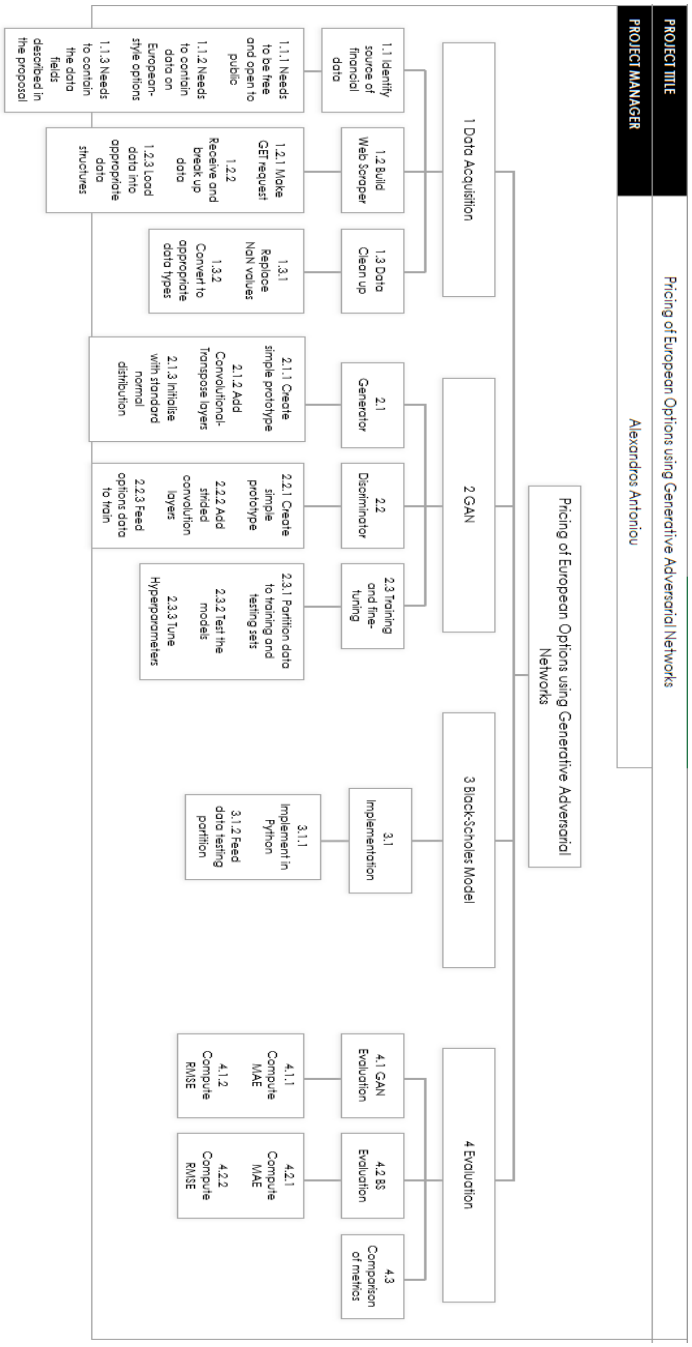

Figure 1Work Breakdown Structure

Project Title	Pricing of European Options using a Generative Adversarial Model						
Name	Alexandros Antoniou	1908855					
Task Name			Apr-20	May-20	Jun-20	Jul-20	Aug-20
Data Acquisition							
	Identify financial data source						
	Build web scraper						
	Clean data up						
GAN							
	Generator						
		Create simple prototype					
		Initialise with standard distribution					
		Add Convolutional-Transpose Layers					
	Create discriminator prototype						
		Create simple prototype					
		Feed options data to train					
		Add strided Convolution layers					
	Training and fine-tuning						
		Partition data to train and test sets					
		Test the models					
		Tune Hyperparameters					
Black-Scholes							
	Implementation						
		Code in Python					
		Feed testing data					
Evaluation							
	GAN						
		Compute MAE					
		Compute RMSE					
	Black-Scholes						
		Compute MAE					
		Compute RMSE					
Report Writing							

Figure 2Gantt Chart