Financial Market Modeling Through Deep Learning Neural Networks and High-Dimensional Data Embedding

Alexander Geiger *Department of Electrical and Microelectronic Engineering, Rochester Institute of Technology, May 17, 2017*

Abstract*— The following paper discusses the methodology behind modeling financial systems through deep learning neural networks. The goal, utilize deep learning to optimize portfolio returns by analyzing time series patterns. Concepts from prior works include advanced feature abstraction techniques such as factor analysis and Principal Component Analysis. New concepts introduced in the paper include network performance optimize strategies, an introduction to relativistic systems and a universal system for scalable deep learning neural network designs with built-in feature derivation. The primary application of the machine learning techniques covered in the following paper will utilize financial datasets provided by the Bloomberg terminal and various other reputable sources outlined in section *model overview*: *source outline

I. Introduction

With the onset of affordable high-performance computing. neural networks provide the opportunity for financial firms to utilize information in a way they never thought possible. By combining human intuition with network analytics market analysis has made once qualitative analytics quantitative.

In the following section, we will review the history of financial analysis and where it is headed. Prior to affordable high-performance computing. Financial analysis was divided into two categories, quantitative and qualitative analysis. Quantitative finance gained popularity with the onset of Operations Research.

Operations Research was first introduced during World War II. Depending on the book you read the origins of Operations Research is often debated among scholars. Whether it be the British or American scientists one thing is clear its purpose was to assess the utilization of war materials based on science rather than wit[9]. After the war, Operations Researched gained popularity in the civilian sector by improving productivity and efficiency, adding a new dimension to global markets[9]. It wasn't long before the finance industry caught on.

In 1951, Peter Whittle revolutionized financial modeling in his publication, *Hypothesis Testing in Time Series Analysis*[2]. In his paper, Whittle debuted the revolutionary concept of an *autoregressive moving average.* Setting in motion what was to be a new era of financial analysis[2]. Nearly three decades later Robert F. Engle[1] took Peter Whittle's idea to the next level. Engle, in his renowned Nobel Prize winning work, *Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation* in 1982 introduced Conditional Heteroscedasticity to the auto-aggressive moving average[1]. The result, financial market volatility could now model turbulent periods, followed by relatively calm periods[1]. However, Engle and Whittle's work had one critical flaw they both utilized univariate analysis which could not encapsulate the abstract economic factor which governs financial markets.

It wasn't until the advent of affordable high-performance computing; technology and human intuition could work together to quantify what qualitative analysis generalized. Recently, innovative machine learning techniques have brought financial modeling into a new era. Several of the most popularized techniques includes deep learning neural networks, support vector machines, quadratic discriminant analysis, and linear regression[3]. However, current machine learning techniques a dialog between user and network. To establish an effective channel of communication.

The communication I am referring to is comparable to how we communicate with technology today. Whether we are sending emails or buying items online we are communicating with technology to attain an end goal. The Communication between user and network enables a user to inform the network, of the data you are giving it and how to handle it. Effectively simplifying incredibly complex tasks.

For the remainder of the paper, the category of machine learning we will be covering is supervised learning. Supervised Learning consists of five main components: *inputs*, *preprocessing*, *features*, *network designs*, and *output target prediction*.

II. Network Methodology

In the following section, we will introduce how to establish an effective channel of communication between user and network. The motivation, behind the implantation of dialog between user and network, is to reduce human error, better manage large network input data and optimization prediction performance. To begin, let us first analyze the general outline for feature abstraction. As shown in the diagram below we see features are derived from preprocessed raw data.

Figure 2.1

Feature Terminology: *Attribute Introduction*

After the features have been derived they are sent to the neural network to predict a target output. The connection between features, layers and output predictions is critical to network design, development, and optimality. When dealing with high-dimensional datasets, allocating *inputs* to *layers* poses major challenges. To overcome this obstacle both feature's and layers utilize ***attributes***. An attribute is an organization system which enables the user to inform the network ***what*** the data is, ***how*** to handle the data, and ***where*** connections need to be made.

The result, an automated process for feature derivation, and network connections. Attributes are comprised of properties, which act as a labeling system allowing the network to interpret, process, and allocate data throughout the network. Effectively standardize *input* to *layer*, *layer* to *layer*, and *layer* to *output* connection through Attribute Logic.

Attribute Logic allows individual network layers to accept or reject an input feature based on their specific ***attribute properties.*** The attribute properties of layers are inherited by a pre-defined logic based network topology. Once a layer ***receives*** an input feature it assimilates to the attributes. Therefore, the layers develop attributes and are able to connect with other layers, as input features did with them. The result, a standardization of combining complex parallel and series substructures in the design of deep learning networks. Figure 2.1 demonstrates how attribute logic can be utilized for network design.

Attribute Connection Outline

The Attribute is composed of five main properties that govern ***Attribute Logic.*** They are defined as, ***Theme***, ***Expression***, ***Group***, ***Trait, and Feature***. For *user-defined* features, only two properties, ***group,*** and ***traits*** are utilized in network design. For simplicity let us cover the user-defined feature network attribute design process:

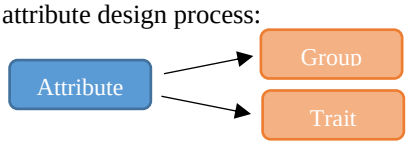


Figure 2.2

The ***group*** determines the origin of raw data. Whereas the **trait** specifies how the data was derived. The shorthand notation for defining attributes of only one group and one trait is denoted as:

$$Attributes \rightarrow \{Group, Trait\}$$

Likewise, the notation for defining attributes with multiple traits $\rightarrow Trait_1, Trait_2$ is written as,

$$Attributes \rightarrow \{Group, Trait_1\ Trait_2\}$$

Where a space separates $Trait_1$ and $Trait_2$. Figure 2.2 demonstrates a simple example of how attribute logic can be utilized to govern the connections between the *input* to *layer*, *layer* to *layer*, and *layer* to *output* connections.

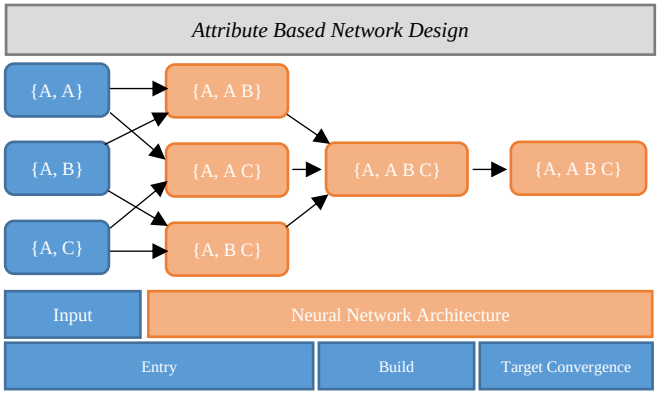


Figure 2.3

Attribute & Feature Automation

For complex networks which utilize automated feature derivation techniques, all attribute properties are used. Properties provide an effective means of data organization and communicating between the network and user. To better convey the development of the neural network design process let us think of it as an assembly line process where data is processed such that it can be fed into a network. The assembly line process is composed of three stages: ***General Preprocessing***, ***Feature Derivation,*** and ***Architecture Design***.

The first stage General Preprocessing begins when key information is abstracted from the raw input data variable names. The abstracted information is then uploaded to the attribute properties. After the attributes have been processed normalization and transformation techniques preprocess the data to optimize network performance.

Once the data has been pre-processed it is now ready to undergo Feature Derivation. Feature Derivation starts when the Macro analysis is applied to the various company episodes. The episodes separate the company's financial data from one another allowing companies to be processed independently and as a group. One of the problems of macro analysis is the statistical calculations utilized in the process produces a large quantity of data. To overcome the problem high-dimensional data embedding is used to make the data's dimensionality more manageable. Once the macro analysis has been applied, time series feature derivation techniques are utilized to analyze data. For a more in-depth look at the methodology of data pre-processing, feature extraction and high-dimensional data embedding please refer to the following section *Feature Derivation and Methodology*.

Lastly, once the attributes have been defined, data has been pre-processed and features have been derived the architecture design process can now begin. The data is then fed into the network through attribute logic. Where the attributes define the connections between the input to layer, layer to layer, and layer to output connections.

Network Design Theory

In ordinary networks, a network is comprised of three layers an *input layer*, a *hidden layer*, and an *output layer*. When dealing with high-dimensional data a single input layer can become problematic. In order to effectively handle high-dimensional data, we discuss how the implementation of attribute logic based topology is implemented to optimize network performance.

To do this deep learning neural network architecture is characterized by three stages which and two hierarchal structures. The hierarchal structures named ***section*** and ***subsection*** are the fundamental building blocks of the network design process. The largest of the bunch sections act as the fundamental building blocks of the stages. Sections are comprised of subsections which intern hold layers filled with neurons. The goal of using an intricate architecture is to parallelized high-dimensional data through attributes to govern data propagation.

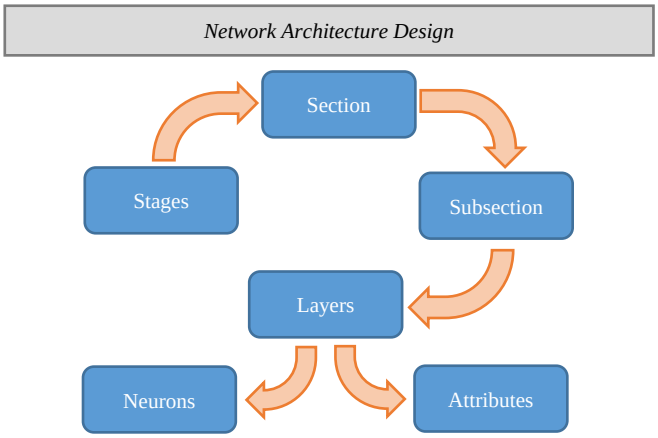


Figure 2.4

The first stage, ***Entry***, is a single and or multi-sectional layer where features and layers make contact. Stage two, ***Build***, is multi-sectional, where layers form connections with other layers. Lastly, stage three, ***Target Convergence***, is a single section comprised of one layer where all layers of the last section's subsection converge to a single layer to predict a target output. The result, neurons are not subjected to high-dimensional data limiting the risk of overfitting and increasing GPU memory. The outline of stage integration into network architecture is outlined in figure 2.4.

Layer Analysis & Neuron Topology

Now that we have effectively established the foundation of how attribute logic is utilized in network architecture. Let us now venture into the layer. A network layer is composed of two key elements Neurons and Attributes.

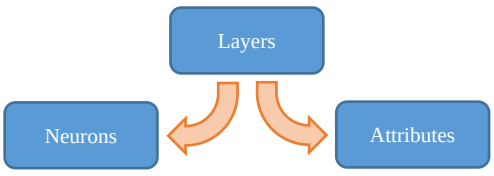


Figure 2.5

Neuron connection and interaction with each other is critical to network optimality. Therefore, neuron topology is used to govern the connections between neurons. The main difference between neuron and layer topology is layer topology is organized. While the topology which governs how neurons connect with one another is random as seen in figure 2.6.

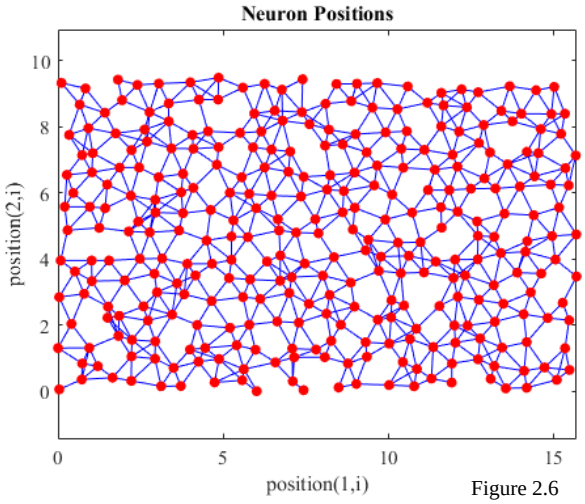


Figure 2.6

Through the utilization of neuron topology and attributes, it is now possible to, optimize the utilization of neurons and automate preprocessing such that deep learning networks can be built from intricate architecture designs like never before. Through the introduction of layer based topology, we are now able to plug high-dimensional data into a network with ease. One of the biggest motivation for the utilization of automating network connections is to allow for automation of feature derivation and integration.

III. Feature Derivation Methodology

Feature derivation and macro analysis techniques. To do this attributes act as the foundation for an interactive network environment. Allowing user intuition to be combined with robust advanced analytics providing a quantitative depth to the user's qualitative intuition. The structure for the automated process is as follows:

Figure 3.1

After raw data is imported into the network environment several Preprocessing techniques are to the data such as transformations and normalization method were used.

Transformation Normalization

For the transformation methods, a Sigmoid Heaviside Laplacian transformation was utilized to filter outliers. The transformation was comprised of three parts and used the mean and standard deviation of the data. The first stage utilized a 1-1 ratio until it passed a number of standard deviations away from the mean, at which point it would jump to stage two or three. Stage two and three utilized a sigmoid function that would act as threshold such that the function would never pass a specified number of standard deviations.

Micro Time Series Analysis

The micro analysis utilizes the preprocessed raw data to derive features. The most popular features include simple exponential smoothing, Holt's exponential smoothing, the generalized conditional heteroscedasticity model, stochastic differential equations and other numerical methods such as derivatives. In order to better understand the methods discussed above let us analyze the equations and their implementation.

First, exponential smoothing is one of the most common methods utilized in financial forecasting. Equation 3.2 gives us the simple exponential smoothing forecasting equation[12]:

$$F_t = \alpha(1 - F_t) \tag{3.1}$$

$$F_{t+1} = \alpha A_t + (1 - \alpha)F_t \tag{3.2}$$

Where F_{t+1} represents the forecasted expected returns of an asset, and α is the smoothing parameter or learning factor. With regards to the learning factor α in simple exponential smoothing. We see the Q-Learning algorithm in equation 3.3 is very similar to simple exponential smoothing. However, in Q-learning, the equation is based on a set of states s and a set of actions a.

$$Q(s,a) \leftarrow (1 - \alpha)Q(s,a) + \alpha[sample] \tag{3.3}$$

Holt took simple exponential smoothing to the next level when he modified it such that it could better handle trends. equations 3.3 - 3.5 give us the Holts exponential smoothing and forecasting equations[11]:

$$F_{t+1} = \alpha A_t + (1 - \alpha)(F_t + T_t) \tag{3.3}$$

$$T_{t+1} = \gamma(F_{t+1} - F_t) + (1 - \gamma)T_t, \ T_0 = 0 \tag{3.4}$$

$$H_{t+m} = F_{t+1} + mT_{t+1} \tag{3.5}$$

Let F_{t+1} be the Level equation, T be the trend equation and H_{t+m} be the forecasting equation. Furthermore, let γ be the smoothing factor or discount factor and α be the smoothing parameter of the Level equation T or the learning factor[11]. Please note Holts original publication in 1957 used different equation notation. The notation has been translated to be more modern[11].

One of the biggest applications of the generalized conditional heteroscedasticity model is assets volatility prediction. Equation 3.6 gives us the Hull representation of the generalized conditional heteroscedasticity model:

$$\sigma_n^2 = \gamma V_L + \alpha\mu_{n-1}^2 + \beta\sigma_{n-1}^2 \tag{3.6}$$

Let γ be your weighting factor to your long-term volatility V_L, α be your weighting factor to your returns μ, and β be your weighting factor on your variance. Lastly, σ_n is the volatility in the market derived from the generalized conditional heteroscedasticity model.

The last micro time series indicator we will be covering is the stochastic differential equation. The stochastic differential equation encapsulates the Holt exponential moving average and the generalized conditional heteroscedasticity model. Equation 3.6 gives us the stochastic equation defined as[12]:

$$ds = \mu Sdt + \sigma Sdx \tag{3.6}$$

Let S be the price of the asset, μ be the drift coefficient defined by the forecasted returns of asset S, σ be the diffusion coefficient current volatility. Next, dt is defined as your time differential and dx is defined as the stochastic probability differential.

Macro Analysis

Another key component to the feature derivation process is Macro-analysis analytics. Macro-analysis utilizes several different feature derivations techniques before undergoing the micro analysis discussed above. These methods include calculating z-scores, standard deviations and variance and other statistical factor to help analyze economic data. Macro-economic data has extracted been extracted across multiple episodes. The data will then be compressed further via high-dimensional data embedding techniques.

High-Dimensional Data Embedding

The main method of high-dimensional data embedding consists of factor analysis and principal component analysis. In figure 3.2 factor analysis is utilized to embed indicator$_1$ five dimensions of data in a two-dimensional surface. How it works is by calculating the maximum likelihood estimate of the factor loadings. The equation for factor analysis is stated in equation 3.7:

$$x = \mu + \Lambda f + e \tag{3.7}$$

Let x be a vector of observed variables, μ be a constant vector of means, Λ be a matrix of factor loadings and e be a vector of specific independent factors. In order to ensure the data embedding provided by the factor analysis is reliable. Algorithms analyze the covariance matrix of that observed data x. Equation 4.8 is used to calculate the covariance of x.

$$cov(x) = \Lambda\Lambda^T + cov(e) \tag{3.8}$$

When using factor analysis, one of the most critical elements to is understanding is the difference between rotated and unrotated data. Unrotated applies equal weights on all of your variable components. Conversely, the weights of the variable loadings can differ in rotated factor analysis. An example of rotated factor analysis is represented in figure 3.2.

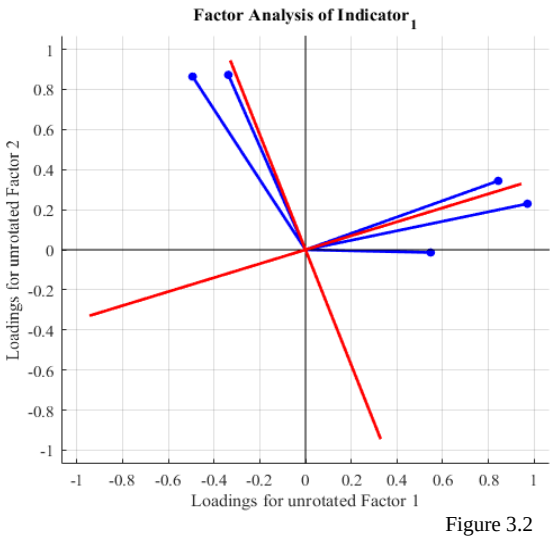


Figure 3.2

IV. Portfolio Optimization Methods

The portfolio optimizing strategy utilizes a portfolio of two assets and cash. The assets are derived from stocks by using an inverse correlation coefficient maximization function. Such that given assets A and B:

$$fmax = -\frac{cov(A,B)}{\sigma_A \sigma_B} \tag{4.0}$$

One the assets have been derived the next step is deciding on how much is to be invested at the risk-free rate given when given a maximum portfolio risk tolerance $\sigma_{portfolio}$.[12] To solve for the risk tolerance, we will utilize the risk tolerance equation:

$$\left(\sigma_{portfolio}\right)^2 = a_1x^2 + a_2xy + a_3y^2 \tag{4.1}$$

Where the x and y represent the percentage of portfolio capital allocated to the assets[12]. The three coefficients a_1, a_2 and a_3 are defined by:

$$a_1 = (\sigma_A)^2 \tag{4.2}$$

$$a_2 = 2\rho_{AB}\sigma_A\sigma_B \tag{4.3}$$

$$a_3 = (\sigma_B)^2 \tag{4.4}$$

Let μ_A be Assets A's forecasted return, σ_A be A's the forecasted variance, μ_B be B's forecasted return, σ_B be B's forecasted variance and ρ_{AB} be the correlation between assets A and B. In order to solve for the portfolio capital allocation of the assets. A maximization function is utilized to forecast asset returns the portfolio[12]. We also introduce a new variable z which represents the capital invested at the risk-free rate of return.

$$f_{max} = rz + \mu_1x + \mu_2y \tag{4.5}$$

$$1 = z + x + y \tag{4.6}$$

Combining the constraint equation, and the maximization equation we are able to produce the following equation which must be maximized[12].

$$f_{max} = r(1 - x - y) + \mu_1x + \mu_2y \tag{4.7}$$

$$f_{max} = r - rx - ry) + \mu_1x + \mu_2y \tag{4.8}$$

$$f_{max} = r + x(\mu_1 - r) + y(\mu_2 - r) \tag{4.9}$$

Now that we have derived all of our formulas we know must now solve for solve for the variables μ_A, μ_B, σ_A and σ_B. First, to calculate the forecasted expected returns we will use Holts exponential smoothing which was covered in section[11]

$$H_{t+m} = F_{t+1} + mT_{t+1} \tag{4.10}$$

Next, to forecast the derived asset's volatilities σ_A and σ_B a generalized conditional heteroscedasticity model is used. The general formula for calculating the generalized conditional heteroscedasticity model is seen in equations[1]:

$$\sigma_n^2 = \gamma V_L + \alpha\beta\sigma_{n-1}^2 + \beta\sigma_{n-1}^2 \tag{4.11}$$

For more information on the derivation process of the generalized conditional heteroscedasticity model please refer to section feature derivation methodology.

To better understand how portfolio optimization can be utilized let us calculate the maximum capital allocation to assets A and B. Given the risk-free rate is 0.01, Asset B has a forecasted return $\mu_B = 0.05$, volatility $\sigma_B = 0.2$ and asset A has a forecasted return $\mu_A = 0.1$, volatility $\sigma_A = 0.4$ with a correlation to asset B $\rho_{AB} = -0.5$. The following graph can be generated by solving the constraint problems. The solution is provided via figure 4.1 below:

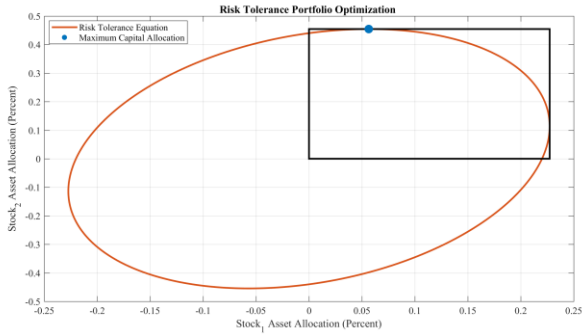


Figure 4.1

V. Delta Hedging

Delta Hedging Overview

One of the most interesting and effective ways to collect the risk-free rate is through delta hedging. In order to better understand the mechanisms behind delta hedging let us price an option which has a payoff in (4.1):

$$P(S) = \max(0, S(1 - S)) \tag{5.1}$$

Furthermore, the asset price derivative S can be modeled by the stochastic differential equation outline in (4.2) [12]:

$$ds = \mu S dt + \sigma S dx \quad (5.2)$$

Let, dX is your probability component, and dt be the amount of time it takes to update your portfolio[12]. To solve for the partial differential equation and derive the option price derivative dV to delta hedge the asset we use Ito's lemma (4.3)

$$dV = V_t dt + V_s dS + \frac{1}{2} V_{tt}(dt)^2 + \frac{1}{2} V_{ss}(dS)^2 \quad (5.3)$$

Since, the dt is considerably we can simplify the equation by assuming $\frac{1}{2} V_{tt}(dt)^2$ converge to zero

$$\frac{1}{2} V_{tt}(dt)^2 \to 0 \quad (5.4)$$

Next, we can now substitute the stochastic differential equation in (4.1) into Ito's lemma to get (4.5) before it is simplified our final result for dV (4.6) [12]:

$$dV = V_t dt + V_s(\mu S dt + \sigma dX) + \frac{1}{2} V_{ss}(\mu S dt + \sigma S dX)^2 \quad (5.5)$$

$$dV = \left(V_t + V_s \mu S + V_{ss} \frac{(\sigma S)^2}{2}\right) dt + V_s \sigma dX \quad (5.6)$$

Now that we have solved for dV and dS we can now utilize delta hedging our asset by setting the change of portfolio dΠ equal to the risk-free rate of return in equation (5.7).

$$d\Pi = V - \Delta(dS) \quad (5.7)$$

$$d\Pi = r\Pi dt = rV - r\Delta(dS) \quad (5.8)$$

Since the objective of delta hedging is to eliminate risk in the market we set Δ equal to V_s in equation (4.9) to minimize risk[12].

$$\Delta = V_s \quad (5.9)$$

Once we substitute in V, dS, & Δ into our portfolio hedging strategy in equation (4.10) the algebra can be boiled down to (4.11):

$$d\Pi = \left(V_t + V_s \mu S + V_{ss} \frac{(\sigma S)^2}{2}\right) dt + V_s \sigma dX - V_s(\mu S dt + \sigma dX) \quad (5.10)$$

$$d\Pi = V_t + V_s \mu S dt - V_s \mu S dt + V_s \sigma dX - V_s \sigma dX + V_{ss} \frac{(\sigma S)^2}{2} dt \quad (4.11)$$

Lastly, it is clear to see we can now derive the solution by setting the two portfolio hedging equations dΠ equal to each other defined in (4.12):

$$rV - rV_s = V_t + V_{ss} \frac{(\sigma S)^2}{2} dt \quad (5.12)$$

To model the partial differential equation several methods including implicit, explicit and the Crank method are all applicable[12]. Please note some methods handle boundary conditions better but there are ways for manually adjusting your matrices for an optimal solution. Form the derivation it is clear to see that the risk-free rate of return can be derived via a stochastic differential equation.

VI. Benchmark & Procedure

Benchmark Overview

For the performance evaluation of an attribute based deep learning neural network a benchmark will be utilized. of The benchmark will be a standard network design, consisting of an input layer hidden layer and output layer. The performance evaluations will entail data training results such as overfitting, and data correlation. Whereas the performance evaluations will be derived from portfolio analysis[12]. The portfolios will be derived from the hedging strategy discussed in section portfolio optimization methods.

Portfolio performance will be based on the returns generated by each portfolio. The portfolio will also have the right to collect the risk-free rate of return on cash which has not been invested into assets. For more information on collecting the risk-free rate in the market please refer to section Delta Hedging[12]. The section provides an in-depth look into the partial differential equations which govern Delta Hedging.

The trading will be conducted through the implementation of a portfolio and fund class which has the right to buy and sell shares of company stock. The buying and selling of company shares will be dictated by a set of rules which governs total capital allocation to the market, and how much to invest in stocks. The capital allocation is determined by a maximum portfolio risk tolerance $\sigma_{portfolio}$ discussed in section Portfolio Optimization Methods[12]. Second, several Fundamental rules such as only invest in a company that has positive future returns, and maximum percent portfolio allocation threshold are defined in the trading algorithm. With regards to the risk-free rate of return *Delta Hedging* company returns will not be derived from shares of stock. Rather the risk-free rate of return will be set arbitrarily to 0.02.

Lastly, a benchmark will be utilized in order to better judge the performance of both the baseline and Attribute Network. The benchmark we are using will be the S&P 500 Index returns. The S&P Index is seen as a reputable leading indicator for U.S. equities[7]. For the returns of the S&P 500 Index please refer to S*&P Returns* table below:

S&P Returns

Year	Quarterly Returns				Returns
	Quarter 1	***Quarter 2***	***Quarter 3***	***Quarter 4***	***Total***
2016	0.77	1.90	3.31	3.25	9.54
2015	0.44	-0.23	-6.94	6.45	-0.73
2014	1.30	4.69	.61	4.39	11.39

Calculation Overview

Over 1.5 million raw data targets will be predicted from two main components. The first component will be comprised of features derived from daily stock market data. The daily stock market data will consist of the high, low and closing price of US Equities over the last decade. Second, an array of

company's financial fundamentals which utilize regression analysis. The financial fundamentals will consist of company assets, liabilities, income, and debt.

Next, the universe will consist of all US Equities with a midcap above twenty million dollars. The target output will be the log return of the company's shares which is then converted into a price. For the sources of the raw data please refer to section *Model Overview*: *Source Outline.*

VII. Network Results

Network Overview

Both the Attribute and Baseline networks were trained on over 1.5 million targets. The performance factor was based on a portfolio investment strategy. Both network portfolios utilized the same hedging and optimization techniques. The hedging strategy utilized macro and micro analysis of the data returned by the network. The macro component determined how much of your portfolio should be invested in assets and what percentage of portfolio capital should collect the risk-free rate of return. Please note portfolio capital was allocated to the assets which intern was distributed among the underlying assets.

Portfolio Performance Evaluation

The Attribute Network or Network I outperformed both the baseline network and the benchmark S&P Returns as shown table *Network I returns*:

Network I Returns Table

Year	Quarterly Log Returns				Returns
	Quarter 1	*Quarter 2*	*Quarter 3*	*Quarter 4*	*Total*
2016	8.6508	5.2596	7.2597	13.6336	34.8037
2015	4.6540	3.7583	2.0595	3.6856	14.1574

Conversely, the returns derived from the baseline network significantly underperformed both Network I and the S&P Returns. One of the main reasons for the low returns was the limited allocation of funds to shares of the underlying assets. Rather, the capital was invested in risk-free rate of return. In the section *Discussion* we will give a more in-depth explanation as to why the baseline portfolio underperformed as it did.

Base Line Network Table

Year	Quarterly Log Returns				Returns
	Quarter 1	*Quarter 2*	*Quarter 3*	*Quarter 4*	*Total*
2016	0.7412	0.6259	0.7639	0.7658	2.8401
2015	0.6256	0.5613	0.4071	0.7803	2.3743

Portfolio Optimization

The following portfolio optimization parameters were the same for both the Attribute and Baseline Networks. The first step in portfolio optimization is to determine the amount of capital invested at the risk-free rate, and how much to invest in the shares of the underlying assets. The capital allocation was determined by a maximum risk tolerance $\sigma_{portfolio}$ which was set to 0.1 in the portfolio management class. To determine the macro-economic asset returns μ Holt's exponential smoothing was used. The hyper-parameters for Holts exponential smoothing were a learning rate of $\alpha = 0.45$ and a discount factor of $\gamma = 0.5$. Lastly, the risk-free rate of return was set to $r = 0.02$ per year. For more information on how Holt's equation was utilized please refer to section Feature Derivation and Methodology

Portfolio Optimization Methods.

Next, to forecast the macroeconomic portfolio volatility a generalized conditional heteroscedasticity model was used. The parameters of the model were set to, γ = 0.01, α = 0.075, and β = 0.27. Lastly, the long term volatility V_L was derived from historical data where as μ_{n-1} used the predicted network data which undergone further processing through Holts equation. For more information on how the hyper-parameters of the generalized conditional heteroscedasticity model were calculated, please refer to section *Portfolio Optimization Methods.*

Training Evaluation

The network training preformed significantly better than the baseline network with absolutely no overfitting as seen in figure 7.1. One of the most impressive performance evaluation metrics Network I exceled at was the high correlation of target to target prediction in the data. The results of the training data can be seen in the table Training Data:

Traning Data Table

	Correlation Value		Percent
Type	Network I	Baseline	Data Division
Training	0.9152	0.3113	0.8000
Validation	0.9059	0.3015	0.1000
Test	0.8957	0.3068	0.1000

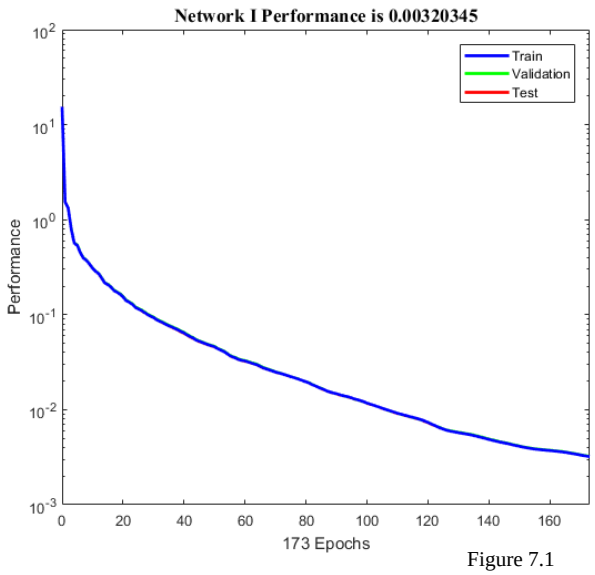


Figure 7.1

Data Normalization

The following methods discussed in *Data Normalization* will only apply to the Attribute Network. Various automated, preprocessing, feature derivation and macro-analysis techniques were applied to the dataset. Preprocessing methods include standard scaling normalization methods while the transformations methods used by the Sigmoid Heaviside Laplacian transformation. The Heaviside function was set to a 3 standard deviations and cap of 6 standard deviations.

Micro Feature Derivation

The following methods discussed in *Micro Feature Derivation* will only apply to the Attribute Network. For time series feature derivation. Holt's exponential smoothing with a learning factor of $\alpha = 0.5$ and a discount factor of $\gamma = 0.5$. Next, stochastics differential equations were another Indicator. To calculate the asset forecasted returns μ of company stock Holt's exponential smoothing was used. The hyper parameters for Holts exponential smoothing used a learning factor of $\alpha = 0.45$ and a discount factor of $\gamma = 0.5$. Next, to forecast a company asset's volatility a generalized conditional heteroscedasticity model was used. The parameters of the model were set to, $\gamma = 0.01$, $\alpha = 0.075$, and $\beta = 0.27$. For more information on the derivation of the stochastic derivation and Holts equation please refer to section feature derivation.

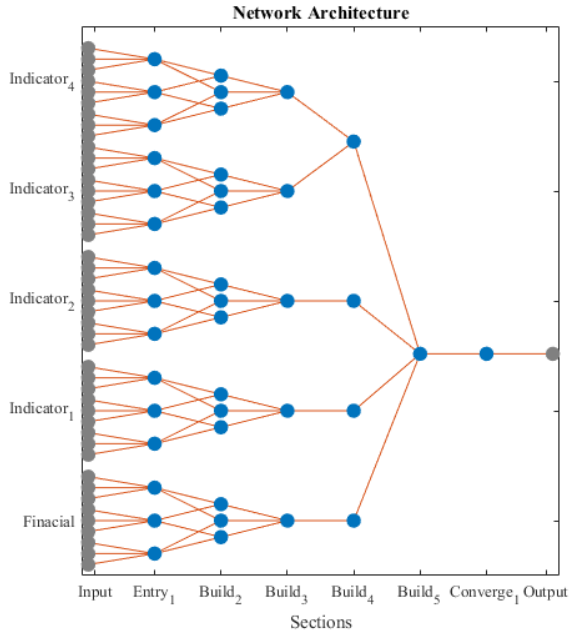
Macro Feature Derivation

The following methods discussed in *Macro Feature Derivation* will only apply to the Attribute Network. Macro-analysis analytics consisted of two stages and utilized several statistical calculations. The statistical calculations included variance, z-scores, and standard deviations to model economic data of multiple companies on a specific day. To relate the macro-economic data back to episode data two indicators were used. The first, denoted as Δ was the difference between the episode value and the mean of the sample data of all the episodes on a specific date. Second, was the delta Δ weighted by the standard deviation. After the Macro features have been derived they underwent further micro analysis. Since The high-dimensionality of the compounded Macro-data via had to be Compressed.

The high-dimensional macro-data was compressed via both factor analysis and principal component analysis through the uses of Matlab built in statistics and machine learning toolbox. A minimum covariance threshold for the loadings was set to 0.65 such that any loading with a correlation less than .65 would be rejected. Principal component analysis was able to be used as well. However, factor analysis utilized a variance maximization rotation. Therefore, since principal component analysis is based on similar variance optimization strategy it's integration into data analysis was discouraged. For more information on the data embedding strategy please refer to the feature derivation section which discusses high-dimensional data embedding strategies.

Architecture Design of Attribute Network

The Network architecture was created using the Network Layer Architecture Class. The overall design utilized a total of 45 inputs, which were imported into the network via a five subsection feature convergence entry stage. Afterwards the entry fed into a five sections build stage followed by a single section convergence stage. Each layer had a maximum neuron capacity of 283 with an 85 neuron coefficient factor per-input as shown in the figure below:



The network and its topology was designed utilized an array of classes and toolboxes. Several of the classes and toolboxes were specially developed for a universal interface between network and user. While other classes and toolboxes were provided by Matlab. For more information on which classes, and toolboxes were utilized table please refer to table 5.1. Note over seventy-five classes were developed in order to make informed market prediction. Therefore, only the most relevant classes and toolboxes will be discussed.

VIII. Discussion

Small Step to Big Things

The beauty of attributes goes beyond a stunning network architecture. Attributes open up a channel of communication between user and network. Allowing human intuition to be combined with network analytics in a way we are already are familiar with. Whether we are sending an email to a friend or buying items on line we are communicating with technology to accomplish an end goal.

Through the use of communication between user and network we are creating an environment which encapsulates all of the complexities of behind building and training networks. Allowing for feature to be derived to be derived with ease and the integration of high-dimensional data in a network to be even easier.

Experiment Discussion

When comparing the baseline network and Network I there is no questioning what the power of an attribute based system can do for you. Through predicting returns to a significantly higher degree of accuracy Network I was able to be easily embedded into complex hedging strategies to produce optimal results, while balancing risk.

Conversely, the baseline network failed to produce reliable results. The portfolio hedging strategy deemed the derived assets to be volatile and unreliable. The high volatility and unreliable returns increased the portfolio capital invested at the risk-free rate of return rather than in the shares of the underlying assets.

As for the training with regards to overfitting both the Attribute based network and the baseline network performed very well. However, if you look closely at the Network I Training graph you will see that the results could have even more optimal as the training curves had not fully converged. The reason the Network's I training was stopped early was because it had been training for nearly six days and I needed to preform portfolio analysis.

Lastly, the correlation between the target inputs and outputs was significantly higher for Network I than the baseline network. The reason for the change in correlation between the target outputs and predicted target outputs was because of the feature derivation, and the depth of the network. Network I utilized an array of feature derivation techniques covered in section Feature *Derivation Methodology.* Whereas the baseline network utilized no feature derivation.

Portfolio Optimization Discussion

For future portfolio's the optimization parameters could undergo further optimization. The hedging strategy did not allow for a significant amount of risk. As a result, both Network I and especially the Baseline Network were restrained to significant risk. One of the main motivations behind reducing volatility was because volatile systems the system needs to overcome a barrier of $\frac{2x}{3}$ percent barrier of non-log returns where x was the last percentage drop. Such that if the returns are not long returns, and the shares follow the pattern:

$$\frac{-3}{6}, \frac{1}{2}, \frac{-3}{6}, \frac{1}{2}, \dots, \frac{-3}{6}, \frac{1}{2} \to 0$$

$$\frac{-1}{3}, \frac{1}{2}, \frac{-1}{3}, \frac{1}{2}, \dots, \frac{-1}{3}, \frac{1}{2} \to x$$

the original investment will converge to zero. Furthermore, where x is the original investment. Therefore, through the following series it is evident the risk threshold should remain low.

IX. Conclusion

In conclusion it is clear to see how an attribute based network provides a new level of performance. When compared to too a standard neural networks the attribute based network dominated the performance evaluation of a two-year portfolio by accumulating a total return of 63.9611 percent Vs. the standard network which only produced a 5.3500 percent return.

However, the attribute has a long way to go. Moving forward furthering the development of new and improved feature derivation techniques is critical to improving performance. More so since the attribute technology has embedded a network inside an environment when will the network use features derived from another networks. Such that the network itself become what the perceptron was to the modern layer.

Next, how can you better combined classification and regression to optimize the returns results. Where the ridged structure which segregates classification from regression becomes one in the same.

Finally, how can we create an easy to use interface to analyze results and better communicate with data rather than through a more interactive user interface. An interface which has the ability to make new AI technology affordable and accessible. By doing so we can replace operations research with Machine learning an AI technology. Adding a degree of value operations research could never provide by quantifying type of qualitative data operations research never could.

X. Related Works

Related works consists of holts exponential smoothing analysis which has already been discussed section feature derivation here we will reflect on it. Holts exponential smoothing is very interesting because it utilizes a discounting factor. Second, for the Idea behind the implementation of principal component analysis was provided to me by X. Zhong, D. Enke. However, the paper he utilized a somewhat of abutted approach when deriving his features through principal component analysis. Next, one paper I did find very interesting was Itamar Arel and Derek Rose and Robert Coop on scalable deep learning networks. In many ways the attribute system was inspired by their work but completely different.

XI. The Motivation Behind The Paper

The driving force behind the neural network design discussed in the paper is the concept of an **abstract relativistic systems**. The Abstract system was Inspired by the Lorentz transformation[6]. First introduced in Einstein's revolutionary December of 1916 publication, *Relativity: The Special and General Theory*. Einstein described how the Lorentz transformation[6] utilized a co-ordinate system which allowed the same event to be localized to two frames through

coordinate transformation. The result, distances and velocities can be calculated from different frames of reference even if an object is moving close to the speed of light[6]. The main conceptual difference between the Lorentz transformation[6] and relativistic systems is the idea behind the co-ordinate system. A Relativistic System utilizes the co-ordinate system to quantify the energy potential between the components of a system rather than a distance or velocity. Therefore, to build a relativistic system you must derive features from the energy discrepancies between a system's individual component's frame of reference. The derivation of the features will be discussed in section, *Feature Derivation.*

A Tangible Example of a Relativistic System:

For a tangible example of a relativistic systems let us analyze the socio interaction of person {P} in two scenarios {A, B} where person {P} exists in two different friend group's {A, B} respectively. To make it fun you have the right to buy or short ten percent of person {P} in the two scenarios {A, B}.

In scenario {A}, person {P} exists in group {A}. Where a completive academic drive to be the best and excel is at the heart of the interdependent relationships that bind the group together. Conversely, in scenario {B} person {P} exists in group {B}, where individuals are driven to do the minimum amount of work and still be able to do leisurely activities. As a result of group dynamics it is highly probable person {P} in scenario {A}, has a higher potential for growth relative to his state in scenario {B}. Therefore, person {P} in scenario {A} would be the optimal buy and person {P} in scenario {B} would be the short.

When dealing with financial systems, the value of a company with respect to its competition, acts in much way the same way as the socio relations between individuals and their friend group. To exploit the nature of relativistic relationships in financial systems, a return on an investment is driven by a company's reversion to its mean **relativistic value.** Relativistic value is the energy potential derived by a components frame of reference relative to other components in a system. Lastly, the value of a company can be broken down into four main components, Macro Data, Micro Data, Micro Relativistic Data and Macro Relativistic Data.

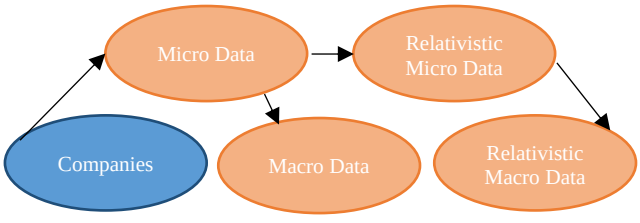


Figure 11.1

XII. Data Sources

The Following section will provide tables which outline the souses by which the data was collected. The majority of data was provided through the Bloomberg terminal. Two tables will provide information of company financials and fundamentals. While the other will cover how the Macro-Economic data was derived.

Company Financials		
Type	***Source***	***Time Period***
Equity	Bloomberg	Quarterly
Liabilities	Bloomberg	Quarterly
Short Term Debt	Bloomberg	Quarterly
Long Term Debt	Bloomberg	Quarterly
Outstanding Shares	Bloomberg	Quarterly
Income	Bloomberg	Quarterly
Price to Book	Bloomberg	Quarterly
Close Price	Bloomberg / Yahoo / Finviz	Daily / Daily / Daily
High Price	Bloomberg / Yahoo	Daily / Daily
Low Price	Bloomberg / Yahoo	Daily / Daily

Macro-Economic Data		
Type	***Source***	***Time Period***
US Debt	Bloomberg	Quarterly
Global Interest Rates	Bloomberg	Quarterly
Oil Production	Bloomberg	Quarterly

XIII. Classes And Toolboxes

The following section will provide a table outlining the main classes and toolboxes which were used to make the paper possible. Since over 65 personal classes were written for the paper and several others were provided by Matlab. I will only include the most critical classes.

Class & Toolbox Outline Table 5.1		
Type	***Name***	***Source***
Class	Feature Derivation	Alex Geiger
Class	Network Fields	Alex Geiger
Class	Network Connector	Alex Geiger
Class	Attribute	Alex Geiger
Class	Network Layer Architecture	Alex Geiger
Class	Data Analysis Class	Alex Geiger
Class	Portfolio Management	Alex Geiger
Class	Fund Management	Alex Geiger
Class	Security Class	Alex Geiger
Class	Macro-economic Analysis	Alex Geiger
Toolbox	Logic Analysis Toolbox	Alex Geiger
Toolbox	Data Analysis Toolbox	Alex Geiger
Toolbox	Network Toolbox	Alex Geiger
Class	Network	Matlab
Class	Linear Model Class	Matlab
Toolbox	Neural Network Toolbox	Matlab
Toolbox	Statistics and Machine Learning Toolbox	Matlab
Toolbox	Financial Toolbox	Matlab
Toolbox	Econometrics Toolbox	Matlab

References

[1] Engle, Robert F. "Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation." Econometrica. Vol. 50, no. 4, 1982, pp. 987-1007

[2] Whittle, Peter, "Hypothesis Testing in Time Series Analysis", Almqvist & wiksells, 1951

[3] X. Zhong, D. Enke. "Forecasting daily stock market return using dimensionality reduction"**.** Expert Systems with Applications, 67 (2017), pp. 126–139

[4] Ray Dalio. "II. Debt Cycles." Bridgewater Research Paper, 2015.

[5] Brandimarte, Paolo. "Numerical Methods in Finance and Economics", 2ed Edition. John Wiley & Sons, Inc. 2007.

[6] Einstein, Albert. "Relativity: The Special and General Theory" Methuen & Co Ltd, December,1916

[7] Ira G. Kawaller, Paul D, Koch, and Timothy W. Koch. "The Relationship between the S&P 500 Index and S&P 500 Index Future Prices" Federal Reserve Bank of St. Louis. May 1988.

[8] Itamar Arel and Derek Rose and Robert Coop. "DeSTIN: A Scalable Deep Learning Architecture with Applications to High-Dimensional Robust Pattern Recognition." The University of Tennessee, Fall 2009.

[9] Hamdy, A. Taha. "Operations Research, an Introduction." Tenth Edition Boston Pearson 2017

[10] Sudman Seymour, Mary A. Spaeth. "The Collection and Analysis of Economic and Consumer Behavior Data In Memory of Robert Ferber." Champaign, Illinois, 1984.

[11] Holt, Charles. "Forecasting Trends and Seasonal by Exponentially Weighted Averages. Graduate School of Business, University of Texas at Austin, Austin, TX, USA, volume 20, Issue 1, January-March 2004, Pages 5-10"

[12] Paolo Brandimarte, "Numeric Methods in Finance and Economics." John and Wiley Sons, Inc. 2002, 2ed edition.