Deep Learning to Enhance Momentum Trading Strategies using LSTM on the Canadian Stock Market

by

Rafik Matta, B.A.Sc, University of Toronto, 2011

A Major Research Paper

presented to Ryerson University

in partial fulfillment of the requirements for the degree of

Master of Science

in the Program of

Data Science and Analytics

Toronto, Ontario, Canada, 2019

AUTHOR'S DECLARATION FOR ELECTRONIC SUBMISSION OF A MAJOR RESEARCH PAPER (MRP)

I hereby declare that I am the sole author of this Major Research Paper. This is a true copy of the MRP,
including any required final revisions, as accepted by my examiners.

I authorize Ryerson University to lend this MRP to other institutions or individuals for the purpose of
scholarly research.

I further authorize Ryerson University to reproduce this MRP by photocopying or by other means, in
total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

I understand that my MRP may be made electronically available to the public.

Deep Learning to Enhance Momentum Trading Strategies using RNNs and LSTM on the Canadian Stock Market

Rafik Matta

Master of Science 2019

Data Science and Analytics

Ryerson University

ABSTRACT

Applying machine learning techniques to historical stock market data has recently gained traction, mostly focusing on the American stock market. We add to the literature by applying similar methods to the Canadian stock market, focusing on time series analysis for basic momentum as a starting point. We apply long-short term memory networks (LSTMs), a type of recurrent neural network and do a comparative analysis of the results of a LSTM to a logistic regression (LOG) approach as well as a basic momentum strategy for portfolio formation. Our results show that the LSTM financially outperforms both the LOG and basic momentum strategy, however the area under the curve of the receiver operating characteristic curves show the results do not outperform a random walk selection. We conclude that there might not be enough data in monthly returns for the LSTM in its current configuration.

Key words:

LSTM, RNN, Momentum, Portfolio Formation

Acknowledgements

Completing this research project would not have been possible without the help of many. I would firstly like to thank my supervisor Dr. Alexey Rubstov whose expertise was critical in guiding my efforts in the right direction, especially with determining the topic and the many revisions and iterations of the work.

I would also like to thank Julian Douglass, Head Quant Researcher at RBC Global Asset Management, for his support and feedback throughout this effort which helped me refine my methodology and think through different aspects of the analysis. I would also like to thank Julia Wawrykowicz for her feedback and discussions regarding the phenomena explored herein, as well as my supportive manager Adam McLaurin for giving me all the time I needed to focus on completing this work.

Finally, I would like to thank my parents for their counsel and support and my fiancée for putting up with my weekend absences and constant love and support.

Table of Contents

List of Figures

List of Tables

Introduction

The profession of investment management has been undergoing a radical shift over the last 2 decades with the introduction of index tracking exchange traded funds and a major challenge in the form of fee compression. Passive investing has seen a tremendous rise and as such the methods professional investment managers employ to active management are under pressure to evolve. Predicting stock prices and generating alpha (returns above market return) are at the heart of what professional investment managers do. The evolution of professional investment management saw the adoption of quantitative methods that use linear models on factors or characteristics derived from stock prices. These models have tended to have their short comings and don't always perform well. The use of non-linear methods and other predictive techniques such as machine learning including deep learning has seen tremendous growth in the last decade with the emergence of scalable cloud computing as well as improved hardware. What is old is new again, and many researchers, funds and investment banks are actively exploring and implementing the use of machine learning in their investment and trading processes.

There are two major challenges with applying machine learning beyond simple linear techniques such as regressions which are already quite actively used. The first is the volume and reliability of data and the second is the explicability of the results, especially for the applications of deep neural networks, which is critical for financial regulators when they audit investment managers.

There are three ways to apply machine learning to portfolio management. We can seek insights from non-traditional data sources to form features that would eventually feed a machine learning model, such as attempting to learn investor sentiment from news, social media and regulatory filings. This fits well with what is known as fundamental investing, wherein a portfolio manager and their team focus their analysis on fundamentally important attributes of a company's performance and aim to determine if the market has mispriced the asset. Alternatively, we can use existing historical stock data such as prices, returns and volume and derived financial ratios thereof, and attempt to apply non-linear machine learning methods to them to potentially create new factors that would feed into what are known as multifactor models. Finally, we can use a hybrid approach combining both non-traditional data sources as well as standard financial data.

A commonly used factor in many models is stock price momentum. The momentum effect is well known phenomena in stock price movements. This effect occurs when a stock's price moves up or down, and it generally tends to persist in that path for a period of time relative to its peers. The main objective of this paper is to determine whether it is possible to replicate and enhance stock price momentum using a

non-linear machine learning approach for the Canadian Stock Market. We apply and demonstrate the results of using a long-short term memory (LSTM) reinforcement learning approach in relation to a basic momentum strategy.

Background and Literature Review

This paper contributes to a growing but scarce literature covering a cross section of two separate domains; quantitative finance and machine learning. Further, as most of the literature focuses on the US or the European stock markets rather than the Canadian stock market, the paper will also serve to explore how some of the advancements in applying machine learning to finance might be applied to the Canadian stock market, which has its own set of challenges related to volatility and liquidity.

As this paper will explore if it is possible to enhance a momentum factor using a machine learning, it is important to first establish what momentum is and how it relates to financial analysis. Within stock price prediction, two approaches prevail; either using fundamental analysis, where one looks at important and well established accounting ratios as well as overall macroeconomic conditions to determine if a stock is either over or under valued (i.e. the current value of the stock relative to its intrinsic value), or technical analysis where one assumes that in an efficient market, all information about a stock is already reflected in its current price and the history of that stock's performance is a better determinant of future price growth.

The Efficient Market Hypothesis (EMH) is an outcome of a report by Eugene Fama (1970), which shows that the stock market with appropriate liquidity in a developed economy will have all available information reflected and priced into the current value of a stock. The EMH proposes that without access to information that is not publicly available (i.e. insider information), we cannot achieve outsized risk-adjusted returns. Neither technical nor fundamental analysis can produce risk-adjusted excess returns consistently. Standing in stark contrast to the EMH, one of the most enduring findings of technical analysis, is the empirical finding of price momentum.

Price momentum is the tendency for a stock's price to either rise or fall along a trend over a period of time and is actively used as both a factor in many quantitative strategies as well as a standard part of technical analysis. Jegadeesh & Titman (1993) show that stocks with high past returns over 3-to-12 months continue to perform well over the next few months compared to stocks with low past returns. This stands in contrast to Weak-Form efficiency proposed by Fama (1970), which is one of the three forms of efficiency proposed as part of the EMH. Weak-form efficiency proposes that future prices of stocks are random and already incorporate all information of previous price data, and no form of technical analysis can be used to predict future prices.

More recently, there has been an evolution in technical and time series analysis with the emergence of machine learning techniques being used for price prediction as risk premium prediction. Of

particular interest has been the application of deep neural networks and in particular reinforcement learning to break away from the general trend of applying supervised learning to the asset pricing problem. Fischer (2018), does a survey of the landscape of current literature using neural networks and reinforcement learning, and categorizes the current approaches into three categories; critic-only approach, actor-only approach, and actor-critic. He further expands on the advantages as well as short comings of supervised learning approaches over unsupervised learning approaches.

Building on the work of Jegadeesh & Titman (1993), Takeuchi and Lee (2013) did some of the seminal work in this regard, where they use an autoencoder composed of stacked restricted Boltzmann machines to extract features from the history of individual stock prices and discover an enhanced version of the momentum effect in stocks without extensive hand-engineering of input features. This is an excellent example of unsupervised learning being applied particularly in feature engineering. This approach delivers an annualized return of 45.93% over the 1990-2009 test period versus 10.53% for basic momentum.

Looking at a shorter rebalancing period and investment horizon, Krauss et. al (2017), analyze the effectiveness of deep neural networks (DNN), gradient-boosted-trees (GBT), random forests (RAF), and several ensembles of these methods in the context of statistical arbitrage. They look at creating an equal weighted portfolio consisting of holdings within the S&P 500, over an analysis period from 1992-2015. They find that using machine learning methods, a simple, equal-weighted ensemble consisting of one deep neural network, one gradient-boosted tree, and one random forest produces out-of-sample returns exceeding 0.45 percent per day for a portfolio consisting of 10 stocks, prior to transaction costs. They suggest that their empirical findings pose a severe challenge to the semi-strong form of market efficiency. Semi-Strong form efficiency, also a part of the EMH, says that all publicly available information has already been priced into the value of a stock and neither technical nor fundamental analysis can be used to predict future price movements. This differs from the Weak-Form because in the weak form, only technical analysis is thought to be unable to assist in predicting future price movements. Here the authors have shown that they were able to use machine learning approaches to find pricing inefficiencies over their study period and preform a form of technical analysis to effectively predict price movements. Fischer and Krauss (2018) followed on Krauss et. al (2017), using the same definition of returns from Takeuchi & Lee (2013), to use a LSTM within a similar one-day look ahead period.

Finally, Lim et. al (2019) develop an LSTM based method they call “Deep Momentum Networks”, which builds on time series momentum strategies for futures contracts. They show high out-of-sample

returns relative to traditional methods absent transaction costs, and a marginal increase of 2-3 basis points accounting for transaction costs.

Methodology

Overview

Our methodology is a multi-step approach combining standard data science, machine learning, financial analysis and statistical analysis practices as follows:

1) We first perform an exploratory data analysis to determine appropriateness of the models being applied

2) The raw data is then split into overlapping study periods (by time), composed of training sets (for in-sample training) and trading sets (for out-of-sample predictions) using hold-out cross validation.

3) The classification results of a basic classification with logistic regression (LOG) are used to serve as a benchmark for comparison being a simple and well understood machine learning stochastic method

4) A LSTM is used to predict similar classification results, with an input of a sequence of cumulative monthly returns.

5) A basic momentum strategy following Jegedeesh and Titman (1993) is applied for portfolio construction. This uses cumulative monthly returns and selects for the top and bottom decile stocks at the end of each month creating a long-short portfolio.

6) A similar portfolio construction method is applied to the output of both the LOG and LSTM, ranking on probability of direction.

7) Finally, the financial performance of the basic momentum strategy, LOG, and LSTM are assessed and compared

Data Description, Software and Hardware

Our data is financial time series data focusing on fully adjusted monthly returns for all tradable stocks of the Toronto Stock Exchange (TSX), over a period of 37 years from January 1982 until December 2018, extracted from the Canadian Financial Markets Research Centre (CFMRC) via the Computing in the Humanities and Social Sciences (CHASS) site provided by the University of Toronto. We further sub select the constituents of the S&P/TSX Composite Index for the same time period. A cross-sectional time series constituents matrix was produced using index constituents data from the

Compustat Daily file provided via the Wharton Research and Data Services website (WRDS). We have a total of 5,458 stocks over a time period of 444 months, or 37 years.

Our LOG method is trained using Sci-Kit Learn and Python. The LSTM leverages Tensorflow and Keras via Google Co-Lab and makes use of tensor processing units (TPU) for rapid training. For all other analysis as well as portfolio construction, Python with Pandas is used.

Basic Data Exploration

Not all stocks have price and return data at any given point in time. It is important to know the overall size of the market and have an idea of the selection of distinct tradable stocks at any given point in time, from which we chose our sub-selection when we are either benchmarking against a particular index (such as the S&P/TSX Composite Index) or just creating a portfolio with a selection of those stocks. Many stocks experience corporate actions and events that may cause their stock to become delisted from an exchange. Others might not even have existed at a given point in time. As such, it is important to have a view of how many tradable stocks there are in the data at any given point.

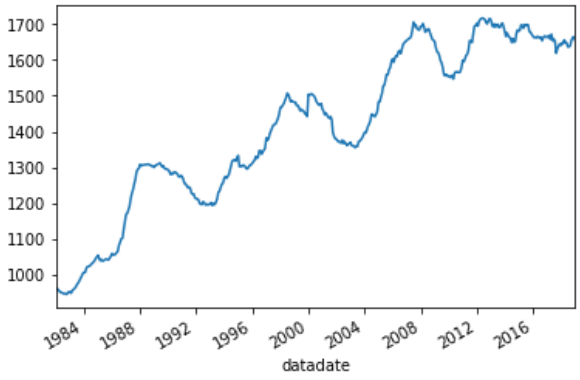

Figure 1. Count of Distinct stocks at any given point in time in the Monthly Data Set.

We see in Figure 1 the trend of number of stocks at any given month that are listed. Macroeconomic events such as the stock market crashes of 2001 and 2009 are apparent and it is clear that many companies became delisted during that period with a rapid upswing in publicly listed companies occurring between 2002-2008 followed again by 2010-2012. This is of particular importance in the formation of our momentum strategy as it has been shown that this can have a significant affect on returns by Eisdorfer (2008).

Example Time Series Plots

As we are primarily concerned with price and return data, it is important to first establish the appropriate statistical view of this data. As a stock's price cannot fall below zero when we are looking at

price data, a lognormal distribution is more appropriate over a normal distribution. Since returns can be negative, a normal distribution would be appropriate for return values. Outliers in our data would be any value well above 3 standard deviations. In some finance literature large anomalous price movements can exhibit this behavior and as such are discarded.

To demonstrate some of the above properties, the first stock in the set with ticker 'A' or Abitibi Consolidated Inc. is chosen and we explore summary statistics as well as visualize what the price and return data look like for the given period. Selecting an individual stock to review is a rational choice given that analysis looks at a particular index (the S&P/TSX Composite Index), which contains a subset of the overall TSX stock market and the constituents of that index change over time. The returns are fully adjusted returns, meaning that they incorporate any corporate actions (such as stock splits), as well as dividends. With momentum strategies, both monthly and daily returns can be used but that entirely depends on the investment horizon. For this analysis, monthly data and therefore monthly returns are chosen.

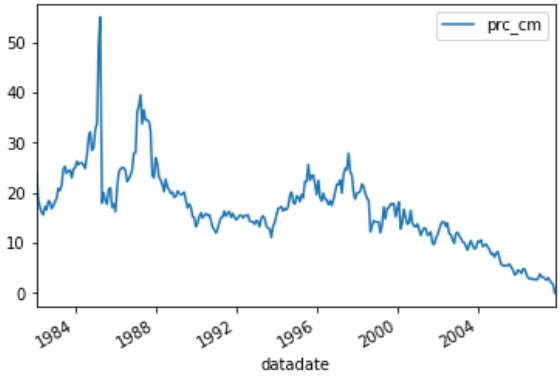

Figure 2. Monthly Price Movement for Abitibi Consolidated Inc.

	PRC_CM	MRET
COUNT	310.00	309.00
MEAN	16.72	0.00
STD	7.77	0.10
MIN	0.00	-0.33
25%	12.65	-0.06
50%	16.38	0.00
75%	20.59	0.06
MAX	55.00	0.46

Table 1. Summary Statistics for Monthly prices of Abitibi Consolidated Inc. for closing price and monthly return.

We visually explore the price trend of this stock in Figure 2. There is an obvious downwards trend with the price. This stock has priced data from Jan 1982 to the Dec 2007. Its price data does not span the entire history and it abruptly ends in 2007. This company experienced a merger and was delisted in 2007.

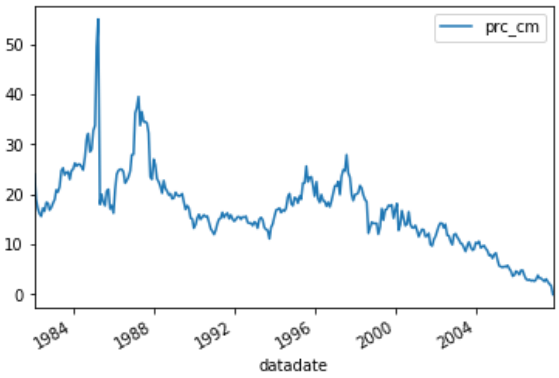

Figure 2. Monthly Price Movement for Abitibi Consolidated Inc.

	PRC_CM	MRET
COUNT	310.00	309.00
MEAN	16.72	0.00
STD	7.77	0.10
MIN	0.00	-0.33
25%	12.65	-0.06
50%	16.38	0.00
75%	20.59	0.06
MAX	55.00	0.46

Table *1* shows summary statistics for closing price ("prc_cm") as well as fully adjusted monthly return("mret"). This stock has a relatively large maximum and minimum price with a large range and a relatively high standard deviation. This indicates that the price movements of the stock were rather volatile. Similarly, for the returns, the expected return value (the mean), is negative and our standard deviation is roughly 10% which is as expected.

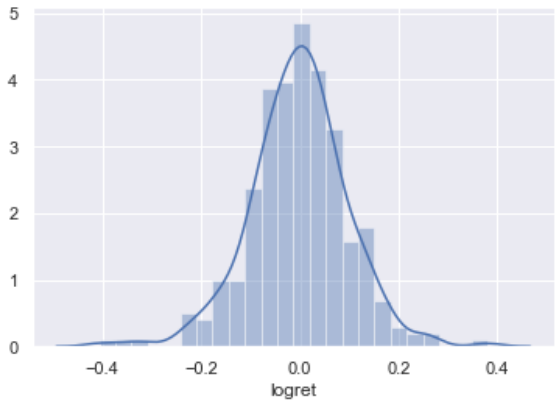

Figure 3. Monthly Return Distribution for Abitibi Consolidated Inc

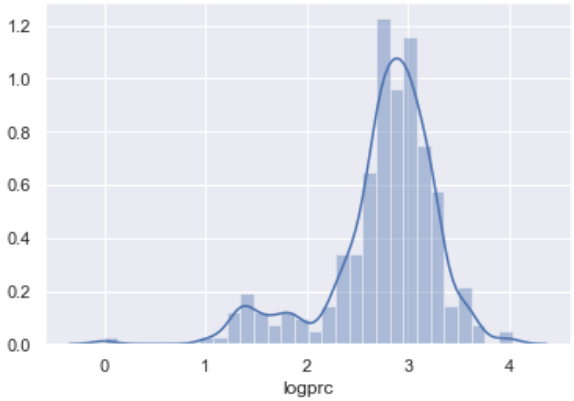

Figure 4. Monthly Price Distribution for Abitibi Consolidated Inc.

In Error! Reference source not found. and Figure 4 we see the distribution price and returns. It is clear that for this stock there is a negative skewness for its price values and only a minor positive skewness for its returns' values. The distribution of returns is closer to being normal than leptokurtic. Generally speaking, the return values of any stock are rarely normal, which is what makes predicting future stock prices challenging.

Experiment Design

As this study is focused on determining if a momentum factor can be enhanced using a deep learning approach and be effectively applied to the Canadian stock market, our experiment design borrows from the work of both Takeuchi and Lee (2013) for the actual experiment setup and Krauss and Fischer (2018) for the construction and implementation of the LSTM Network and comparative benchmarks.

For both the LSTM and LOG methods, we do a rolling window approach to training. Our study periods are overlapping 20-year windows, where our models are trained on 18 years of data, and tested on 2 years of data with the final year (year 19-20), being the year for which the direction of the given stock is being predicted for the following month.

Target Variable Selection

The target variable chosen is the direction of the return of a given stock for the following month. Within our dataset, the fully adjusted return is extracted from CHASS for both our monthly and daily data. Following Takeuchi and Lee (2013) we are solving a binary classification problem. The response variable Y_{t+1}^{s} for each stock s and date t can take on one of two values representing either class. The one-period returns R_{t+1}^{s} of all stocks s in period $t+1$ are cut into two equally sized classes. As Krauss and Fischer (2018) and Takeuchi and Lee (2013) have done, within our setup Class 1 is realized if the one-period return R_{t+1}^{s} of stock s is larger than the cross-sectional median return of all stocks in period $t+1$. Similarly, Class 2 is realized if the one-period return of s is smaller than or equal to the cross-sectional median.

Input Features

As the focus of the study is the effectiveness of LSTM networks, we expand here on what the input features to our network will look like. Following on the approach of Krauss and Fischer (2018), let $P^{s} = (P_{t}^{s})_{t \in T}$ be defined as the price process of stock s at time t, with $s \in \{1, \ldots, n_i\}$ and R_t^s the simple return for a stock s over a monthly period, i.e.,

$$R_{t}^{s} = \frac{P_t^s}{P_{t-1}^{s}} - 1$$

The above is a simplified view of returns, as the returns used are fully adjusted and incorporate corporate actions such as distributions and stock splits, but the essence is captured. For the LSTM networks, for each month t and each stock s, we calculate cumulative returns for each of the prior 12 months and put

them in one large feature vector V of dimension $n_i \times T_{study}$, where T_{study} denotes the number of months in the study period window. For the LOG, we use a single feature of cumulative return for the prior 12 months, calculated as follows:

$$\sum_{t-12}^{t=0} \ln(1 + R_t^s)$$

Cross Validation of Input Data

As this is time series data, we must contend with look-ahead bias. Look ahead bias occurs when data or information that would not have been known or available during the period being analyzed influences or informs the outcomes of a study. Krauss and Fischer (2018) do end up using 5-fold cross validation for their logistic regression model but this is not the case for their LSTM implementation.

To maintain consistency within our analysis and to allow the LOG to serve as a correct benchmark we follow the same cross-validation approach for both the LSTM as well as the LOG. Following Takeuchi and Lee (2013), we use a hold-out instead of a k-fold cross validation as Krauss and Fischer (2018) did with their LOG. Based on the approach for cross validation proposed by Alpaydin (2014) in Chapter 19, this is appropriate. Using the hold-out method, with an appropriate split, we can effectively avoid look-ahead bias.

For each study period, the dataset is divided into a training set (80% of the data or 18 years) and a test set which will encompass 20% of the data (2 years of data, for which the first 12 months are used as input features). Each model is trained on the first subset and then tested on the second.

Model Design and Selection

Logistic Regression as Benchmark Model

Our baseline model is a logistic regression (LOG) which is a standard linear classification model and generalizes well. It is widely used and produces easily interpretable results. By using a LOG model, we can determine the added value of using the much more complex and computationally intensive LSTM. We use the implementation available in sci-kit learn and apply an LBFGS optimizer.

LSTM Network Design

It is important to establish some background on what LSTMs are prior to diving into how they are used within the experiment design. Long-Short Term Memory (LSTM) are a subset of Recurrent neural networks (RNNs) which are a grouping of neural network architectures that embed and leverage

memory in the learning process. RNNs offer the ability to solve the long-term dependency issue, wherein, theoretically they should be able to retain previous information for as long as necessary. LSTM networks offer an advantage as they are explicitly designed for the task of long-term dependency retention. For the sake of brevity, we show in Figure 5 the network topology used followed by an explanation only of the fundamental equations from Krauss and Fischer (2018) . For a more thorough explanation of LSTMs with excellent illustrations, Olah (2015) provides a step by step guide to understand how LSTMs work and their main advantages. The network has the following structure:

• Input layer with 1 feature, x_t and 12 timesteps

• LSTM layer with h_t hidden neurons with a dropout value of 10%

• Output layer (dense layer) with two neurons y using a softmax activation function

As we are following the same network construction, memory cell definition and notation as Krauss and Fischer (2018), the below variables are provided for the readers convenience.

- x_t, input vector at timestep t
- $W_{f,x}, W_{f,h}, W_{s,x^{\sim}}, W_{s,h^{\sim}}, W_{i,x}, W_{i,h}, W_{o,x}, and\ W_{o,h}$ are weight matrices
- $b_f, b_{\tilde{s}}, b_i, and\ b_o$ are bias vectors
- $f_t, i_t, and\ o_t$ are vectors for the activation values of the respective gates
- $s_t\ and\ \tilde{s}_t$ are vectors for the cell states and candidate values
- h_t is a vector for the output of the LSTM layer

At the core of an LSTM is the memory cell unit which is part of the hidden layer within an LSTM network as seen in Figure 6 which shows the basic structure of a memory cell unit as represented by Krauss and Fischer (2018), and how the above variables interact.

As an LSTM effectively retains all prior information, the essential first step is defining the forget gate. The output f_t of the forget gate at timestep t are computed based on the current input x_t, the outputs h_{t-1} of the memory cells at the previous timestep $(t-1)$, and the bias terms b_f. In order to ensure our range of values is between 0 (completely forget) and 1 (completely remember), the softmax function (σ) is used to scale the output of the base equation.

$$\mathrm{f}_t = \sigma(\mathrm{W}_{\mathrm{f,x}} x_t + \mathrm{W}_{\mathrm{f,h}} \mathrm{h}_{t-1} + \mathrm{b}_f)$$

After the forget gate values are calculated, the next step is to determine which information should be added to the network's Cell states (s_t). The Cell state is the current state of the memory cell. In order to

determine the Cell state, the candidate values $\tilde{s_t}$, are computed followed by the activation values i_t of the input gates (again, using a softmax activation function).

$$\tilde{s_t} = \tanh(W_{\tilde{s},x} x_t + W_{\tilde{s},h} \mathrm{h}_{t-1} + \mathrm{b}_{\tilde{s}})$$

$$\mathrm{i}_t = \sigma(W_{i,x} x_t + W_{i,h} \mathrm{h}_{t-1} + \mathrm{b}_i)$$

Then the new cell states s_t are calculated based on the results of the previous two steps with $\circ$ denoting the Hadamard product:

$$\mathrm{s}_t = f_t \circ s_{t-1} + i_t \circ \tilde{s_t}$$

Finally, the output h_t of the memory cells is derived as denoted in the following two equations:

$$\mathrm{o}_t = \sigma(\mathrm{W}_{o,x} x_t + \mathrm{W}_{\mathrm{o,h}} \mathrm{h}_{t-1} + \mathrm{b}_o)$$

$$\mathrm{h}_t = o_t \circ \tanh s_t$$

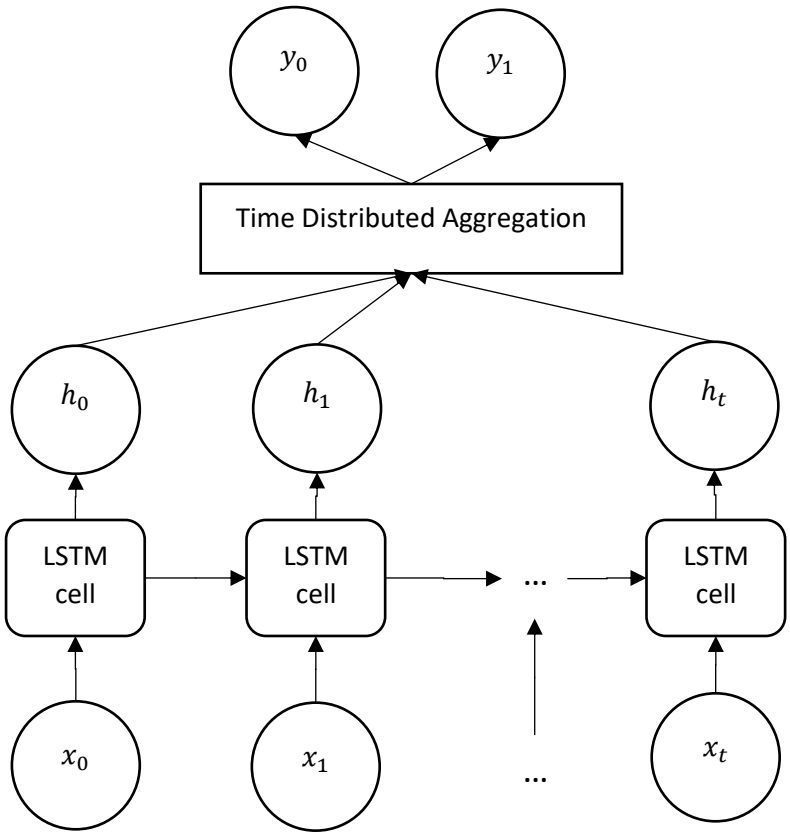

Figure 5. LSTM Network Diagram. Here the network is rolled out (rather than being compacted into vectors)

As this is a classification task, we are aiming to minimize some loss function $L(W, B|j)$. Since we have a binary classification problem, we use the same loss function used by Krauss et. al (2017).

$$L(W, B|j) = -\sum_{y \in O} (\ln\left(o_y^{(j)}\right) t_y^{(j)} + \ln\left(1 - o_y^{(j)}\right)\left(1 - t_y^{(j)}\right))$$

Here we are minimizing a cross-entropy loss function with y representing the output units and o the output layer. This loss function is minimized by stochastic gradient descent.

LSTM Training

We are using the LSTM implementation available in the Tensorflow Keras python library. Similar to Krauss and Fischer (2018), our chosen optimizer is RMSprop. Based on the Keras documentation, RMSprop is seen as a good choice and starting point.

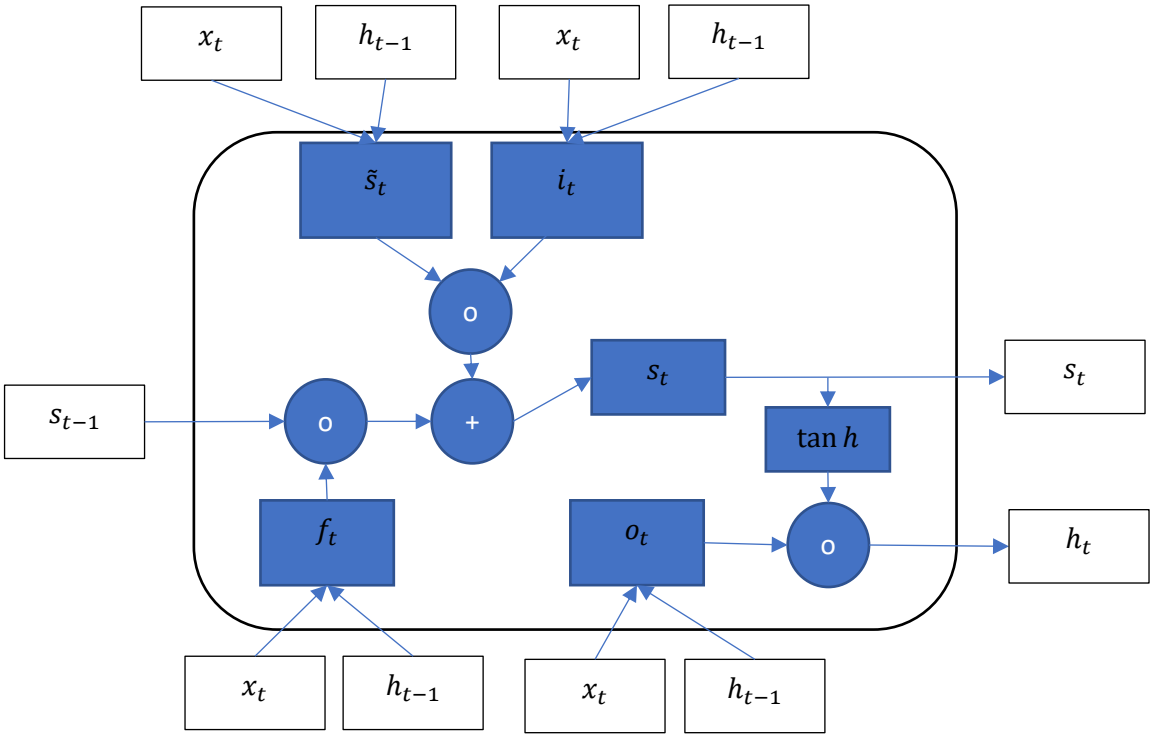

Figure 6. Diagram of a LSTM Memory Cell

One of our tunable parameters and a method to prevent overfitting is regularization. Regularization is the process of regulating network layers to prevent overfitting. The most commonly used approach is drop-out regularization which involves dropping inputs or neurons within a layer to decrease the degrees of freedom. Another possible tunable hyperparameter for training an RNN is the number of

epochs on which to train the network. In this case, an epoch is "one pass across the samples of the first set" (Krauss et al. (2018)). There are diminishing returns at which point the validation loss ceases to be reduced, so the number of epochs is an important parameter to reduce the overall time required to run the analysis. Further, there is a high likelihood of overfitting with a larger number of epochs. A mechanism that is applied to prevent overfitting is early stopping. Early stopping involves splitting the training samples into two sets: one training and one validation set. The training set is used to minimize the loss function by being used to train the network and iteratively adjust its parameters. With early stopping we are aiming to determine when the validation loss ceases to decrease, after which training is stopped and we choose the weights with the lowest validation loss.

Portfolio Construction Strategy

In order to determine the effectiveness of the results of the LSTM against a basic momentum strategy, portfolios of winners and losers are created based on the results of each model, by choosing the top and bottom deciles ranking on a particular criterion for each method. Following Takeuchi and Lee (2013), as this is a classification task where winners are classified in Class 1 and losers in Class 2 based on the probabilities of whether or not they will outperform their cross-sectional median, we rank all stocks each month by their Class 1 probabilities and buy those in the top decile and sell those in the bottom decile. We use a basic equal weighted, or $\frac{1}{n}$ portfolio strategy. This is also known as a naïve strategy and has been shown to perform well for out-of-sample returns by DeMiguel et al. (2007).

For constructing the basic momentum strategy we follow an approach leveraging the python code produced by Qingyi (Freda) Song Drechsler on the WRDS website (Dreschler 2018), which is a reproduction of the approach and SAS code provided by Cici and Moussawi (2004). The authors follow the same procedure for calculating momentum portfolios as Jegadeesh and Titman's (1993), Table 1. We construct a 12-month momentum factor, which means that we are our portfolios are rebalanced every month and are holding period is 12 months.

Performance Evaluation

The fundamental goal of the analysis is to determine if an LSTM will enhance the outcome over a basic momentum strategy. Since LSTM is a predictive classification model, it is also important to assess the performance characteristics of this model such as the accuracy and due to class imbalance, we also look at the area-under-the-curve (AUC) of the receiver operating characteristic (ROC) curves. As such there are two key categories of performance measurements that are considered.

1) Measurements of the predictive capability of LOG and the LSTM as models
2) Financial performance comparison of basic momentum vs. LOG vs. LSTM on key metrics

For the first set of measurements, as this is a two-class classification problem, we will follow Alpaydin (2014) and generate both a confusion matrix measuring accuracy as well as generating the receiver operating curve for each model.

For the second set of measurements, we can look at metrics such as the overall return of the generated portfolios, standard deviation and the Sharpe Ratio. These are all standard ways of assessing the financial performance of a portfolio by looking at the risk adjusted returns.

Results

The results are separated into two sections. The first is focused on standard machine learning discipline looking at how the models have performed as classifiers. The second is focused on the financial performance of the portfolios generated by ranking the probabilities produced by the models against the basic momentum strategy used for portfolio formation. As this work is a multidisciplinary approach, it is important to separate the two concerns and approaches to evaluation.

Classification Performance

As we are dealing with financial time series data in the form of month-over-month stock returns, our models are being trained on a moving window of 20 years. Each model was trained on 18 years of prior data from our holding date t, and tested on two years of out-of-sample data. The input features for each model differ. For the LOG, we use a cumulative return value for each stock over the previous 12 months. For the LSTM, we use a feature vector composed of a sequence of the prior 12 months of cumulative return data for each month, for any given stock s. There is no value in doing a head to head comparison of the performance for the results of both models. We are not aiming to show that one model outperformed the other as a machine learning model or at its particular classification task, but rather the general goodness of fit for both models with their respective data and features.

We see in Table 2 accuracy results of each model. As the training data was a moving window of 18 years and the models were always tested on the following two years, we generate a time series of 17 years of out-of-sample data from Jan 2001 to Dec 2018. In effect, this means we have 17 models trained along different windows of time. On average, the accuracy of the LSTM was generally poor, with some years performing better than others. For the LOG, on average the accuracy is significantly better for it at its classification task, but it still underperformed for a good number of the years out of sample.

A receiver operating characteristic curve helps us determine the diagnostic ability of a binary classifier. We plot the true positive rate against the false positive rate to find the learning rate (LR in the figures), and we use it to determine the appropriate threshold for the classifier. Our chosen threshold when plotting the curve is a greater than 50% probability that the direction is 'up' (which is our class 1 or positive choice). The area-under-the-curve (AUC) is used as a visual measurement to determine the performance of the classifier, where an AUC greater than 0.5 is expected. The ROC curves for the 17-year period are shown in Figure 7 for the LSTM and Figure 8 for the LOG model respectively. For the LSTM, the best performing model and year is 2012. Beyond that, the model generally has an area-under-the curve (AUC) of approximately 0.5, indicating that the model barely performed any better than a random

walk selection. With the LOG model, the results are substantially different year over year. The best performing year was 2014, with many other years performing very poorly with an inverse ROC curve. Adjustments can be made to the models to improve their fit which will be further discussed in future work. In their current forms, the models do poorly as classifiers. Given the scarcity of data, this is expected with a neural network approach such as the LSTM. With the LOG, this is simply an indication that there is little predictive power in cumulative returns using a LOG setup.

YEAR	LSTM ACCURACY (%)	LOG ACCURACY (%)
2001	52.16	54.14
2002	46.22	55.11
2003	61.49	59.58
2004	54.80	80.45
2005	56.30	71.79
2006	53.58	76.23
2007	46.80	68.81
2008	42.72	32.29
2009	61.87	41.91
2010	64.03	91.13
2011	48.78	70.73
2012	55.14	50.67
2013	55.51	69.70
2014	55.29	80.30
2015	45.09	43.43
2016	58.53	68.46
2017	51.70	72.24
2018	51.70	53.84

Table 2. Accuracy for LSTM (a) and LOG (b) for out-of-sample data

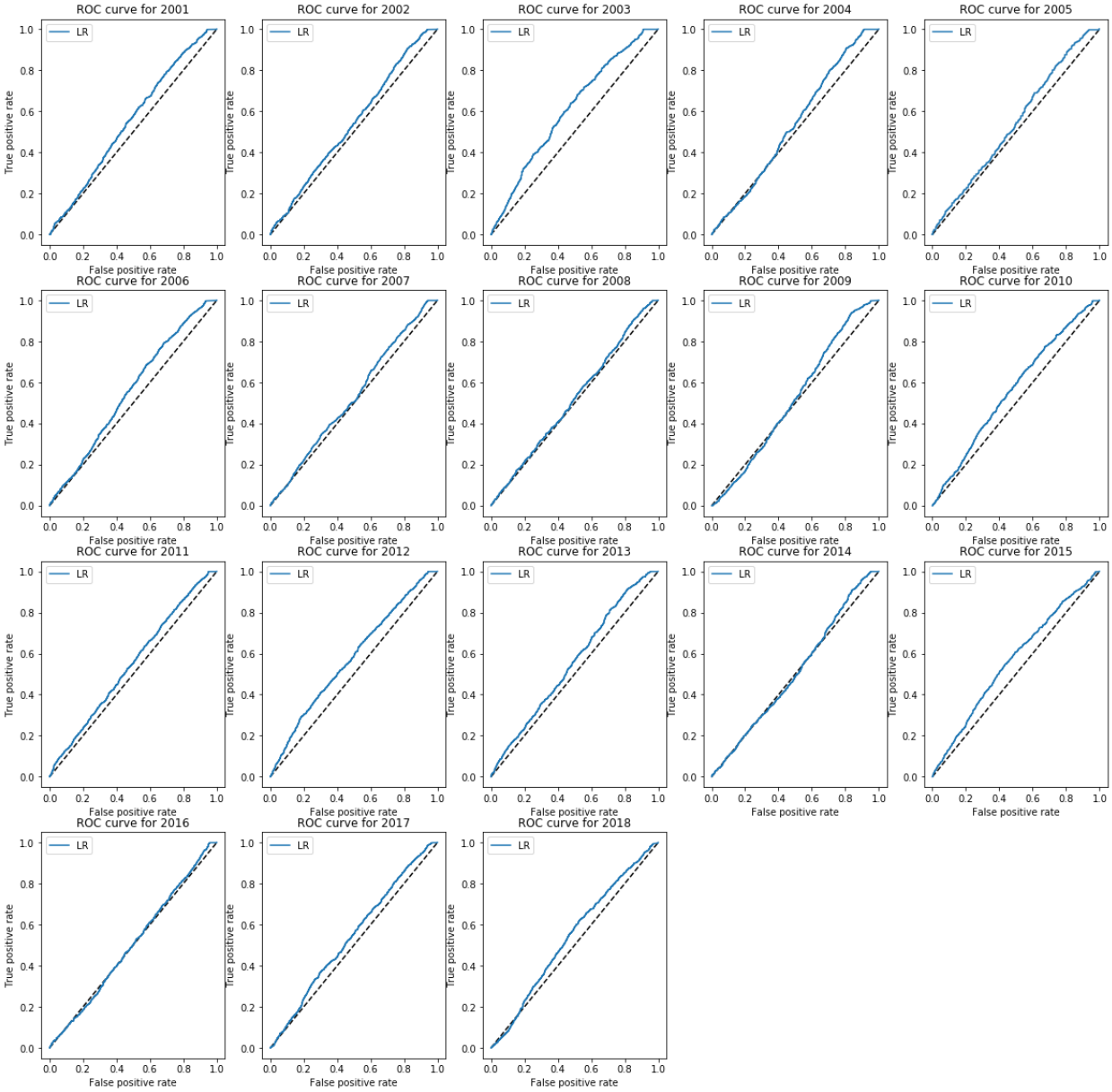

Figure 7. ROC Curves for 17 years of LSTM Model

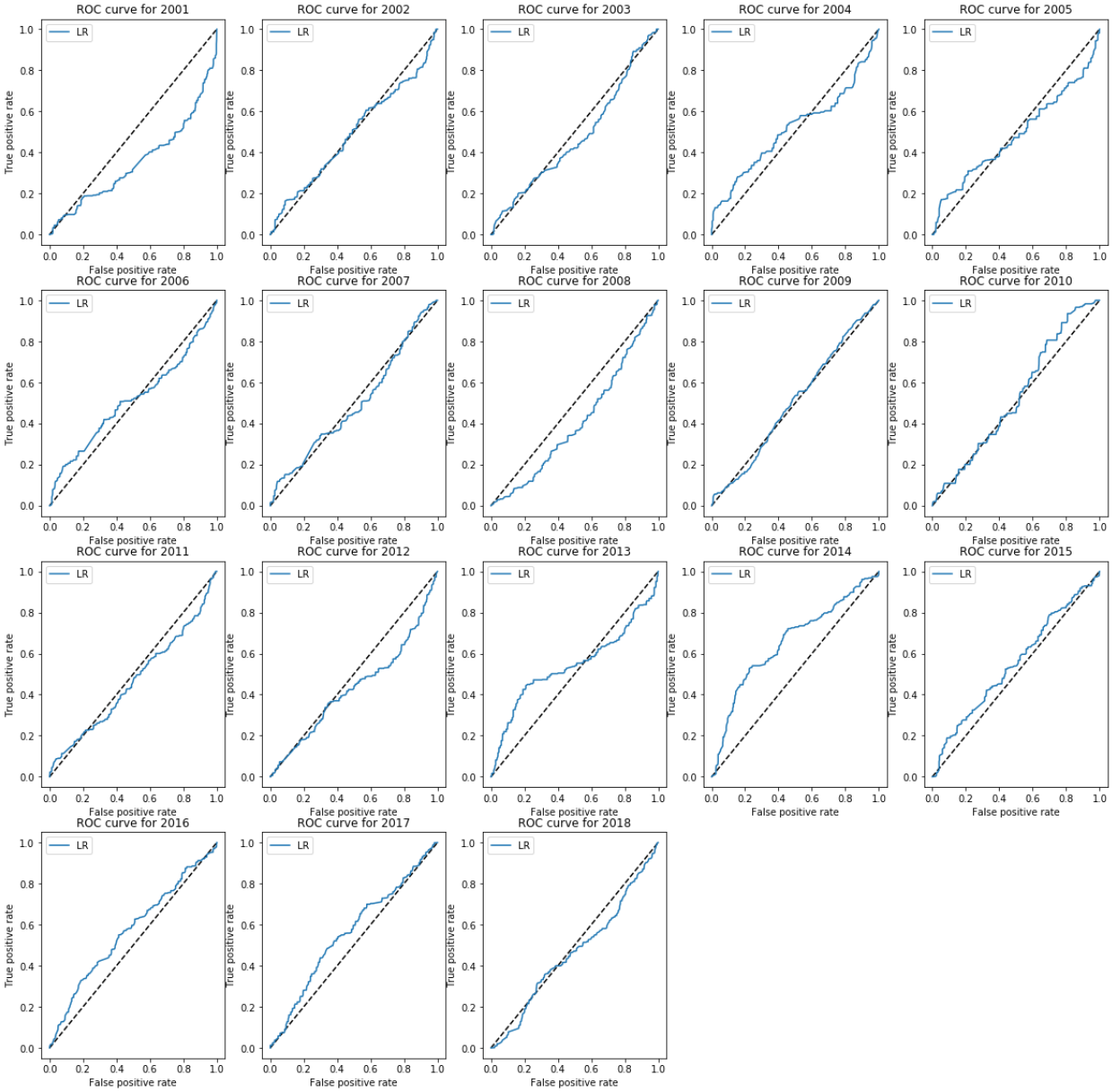

Figure 8. ROC Curves for 17 years of LOG Model

Financial Performance

Focusing on financial performance, the experiments were designed to allow for a head to head comparison of the 3 different techniques. Each technique used the same portfolio formation method, but had a different criterion for selection, as listed below:

- Basic Momentum - ranking on cumulative returns of prior 12 months
- LOG - ranking on probability of up direction, based on cumulative returns of prior 12 month
- LSTM - ranking on probability of up direction, based on a sequence of the previous 12 months of cumulative returns for each month

The portfolio formation strategy is equal weighted portfolios, selecting from the top and bottom deciles for the given period. The number of potential holdings is variable, given that the main criterion for selecting a stock is membership within the S&P/TSX Composite Index prior to the test period. Something not being accounted for here is transaction costs or turnover, which have the potential to alter the results drastically. This is further discussed in the Conclusion and Future Work section.

We can see the overall performance of the portfolios in Table 3, Table 4, Table 5. The highest average return for the long-short portfolio comes from the LSTM produced portfolios, as well as the highest Sharpe ratios. Given the overall poor fit to the data shown for the LSTM in the previous section, we cannot confidently explain the results for the LSTM's out performance against the other two methods except to say that it behaves much like a Random Walk approach.

PORTFOLIO	AVG.RETURN(%)	STD (%)	SHARPE RATIO
TOP DECILE (WINNERS)	7.15	21.55	0.33
BOTTOM DECILE (LOSERS)	4.34	32.11	0.14
LONG-SHORT	2.80	26.81	0.11

Table 3. Portfolio Statics for Basic Momentum Strategy Formed Portfolios

PORTFOLIO	AVG.RETURN(%)	STD (%)	SHARPE RATIO
TOP DECILE (WINNERS)	5.31	24.92	0.21
BOTTOM DECILE (LOSERS)	1.00	25.05	0.04
LONG-SHORT	4.31	24.91	0.17

Table 4. Portfolio Statics for LOG Formed Portfolios

PORTFOLIO	AVG.RETURN(%)	STD (%)	SHARPE RATIO
TOP DECILE (WINNERS)	10.26	12.46	0.82
BOTTOM DECILE (LOSERS)	4.86	19.13	0.25
LONG-SHORT	5.40	10.93	0.49

Table 5. Portfolio Statics for LSTM Formed Portfolios

We can see the time series performance of the winners, losers and long short portfolios in Figure 9. Here the outperformance of the LSTM is very evident. The LSTM has a mean return of 10.26 % for the winner's portfolio and 4.86% for the losers, indicating it did a better job at correctly selecting direction. The long-short portfolio of the LSTM was 5.40% vs. 2.80% for the basic momentum strategy and 4.31% for the LOG strategy. Standard deviation (std in the tables above), serves as a measure of portfolio volatility. Focusing on the long-short portfolio, we can see that the standard deviation of the LSTM is lower than both the LOG and basic momentum portfolios (10.93% vs. 24.91% and 26.81% respectively). Sharpe Ratio serves as a metric of determining risk adjusted returns. Sharpe Ratio is calculated as follows:

$$Sharpe\ Ratio = \frac{R_m - R_{rf}}{\sigma}$$

where R_m is the expected or mean return, R_{rf} is the risk-free rate of return and σ is the standard deviation of the asset or portfolio. A Sharpe ratio of 1 or above indicates acceptable risk-adjusted returns. For the long-short portfolios, the LSTM formed portfolio still outperforms with a Sharpe Ratio of 0.49 vs. 0.17 and 0.11 for the LOG and basic momentum portfolios respectively. For all our portfolios the Sharpe Ratio would be considered completely unacceptable by most investors.

Finally, in Figure 9 we see a time-series visual representation of the cumulative returns for our long, short and long-short portfolios. It is evident that the LSTM portfolio outperforms the other two. That being said, given the generally poor accuracy and AUC of the ROC Curve shown in Figure 7, we cannot confidently attribute this outperformance to the LSTM model's ability to fit well to the data and learn.

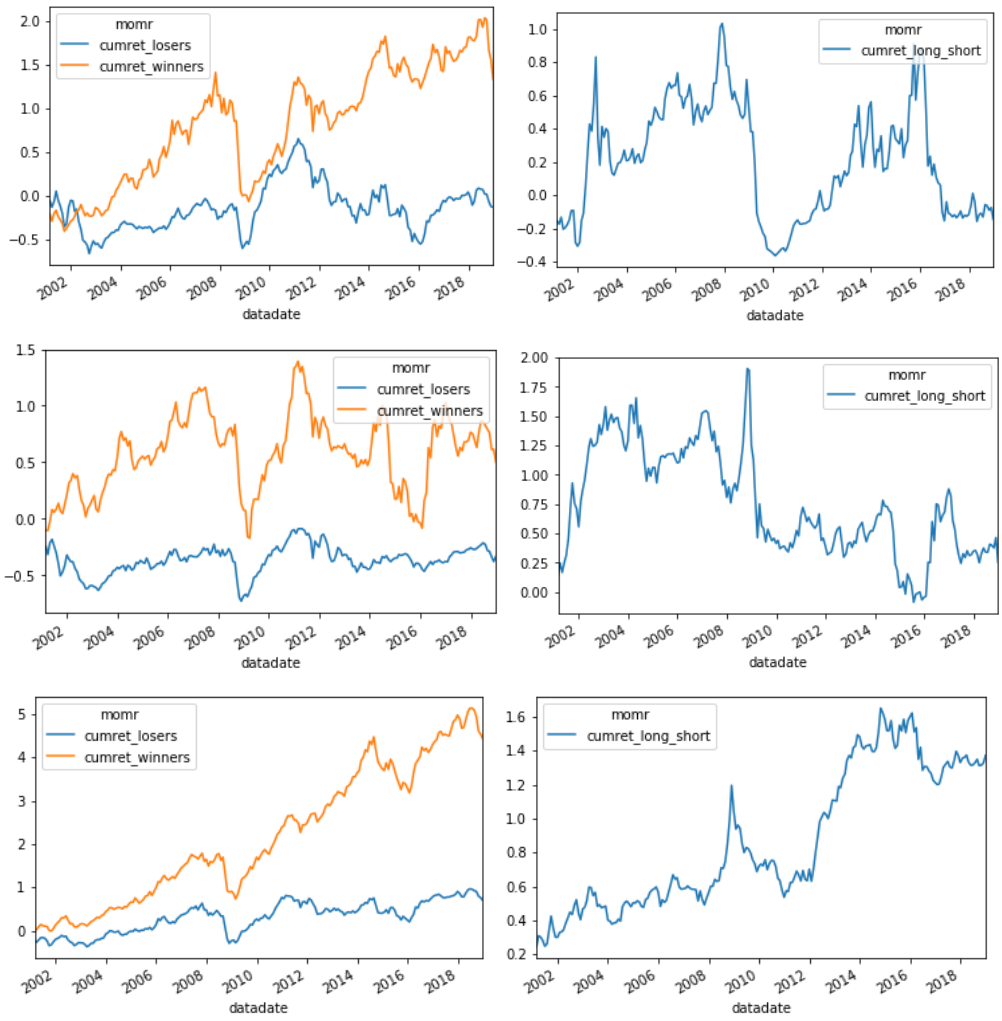

Figure 9. Portfolio Performance for Winner vs Loser Portfolio (Left), and Long-Short Portfolio (right) going from Basic Momentum (top), LOG (middle), and LSTM (bottom).

Conclusions and Future Work

Although the financial performance of the LSTM shows substantially better returns over the out-of-sample test periods, the results are not promising given the AUC of the ROC Curve. Our LSTM model does not perform much better than a Random Walk approach from a predictive perspective. This is also the case for the LOG model. We can expand on this work and take on different approaches to improve the fit, such as hyperparameter tuning of the network specifications for the LSTM. Alternatively, we could try a different study period configuration whereby the model is retrained annually using all available historical data or by applying transfer learning where each year a new year's worth of historical data is added to the model. Both approaches have their benefits and tradeoffs. In any case, it can be correctly concluded that the volume of data, using monthly data, is not sufficient to appropriately train an LSTM to perform well in this prediction task. Another approach might be to incorporate daily data, as Takeuchi and Lee (2013), have done in their analysis, as well as Krauss and Fischer (2018) in theirs where they focused entirely on daily data. One of the major challenges with this for the Canadian Stock Market is the availability of the data. Our study focused on the selection of stocks available in the S&P/TSX Composite Index which limited us to having an earliest date of January 1982. In general, Canadian stock data going further back is hard to obtain and the poor quality of the data also tends to pose a challenge.

One key finding that is not evident in much of the literature is the inclusion of the ROC Curves for our models. Our average accuracy was on par with Krauss and Fischer (2018), but no metric beyond accuracy is offered in that paper. Our inclusion of the AUC of the ROC Curves shows clearly the challenges with this kind of analysis and the detriment in relying too heavily on the results of deep learning models without properly assessing their performance from a machine learning perspective.

From a portfolio construction perspective, an important aspect that was not included in this study is the incorporation of transaction costs and turnover. If a portfolio has high turnover, wherein the holdings are constantly renewed, it would incur a high financial penalty in the form of transaction costs which would reduce overall portfolio returns. This could have tremendous effects on the overall viability of any of the above methods. A simple follow on analysis can incorporate this. Another key follow-on study would be to improve the basic momentum strategy by incorporating one-month reversals which were addressed by Jagadeesh et. al (2001). This would address issues related to the poor performance of the basic momentum strategy during market downturns. It is possible that the LSTM does learn and take into account reversals, which could mean that we can achieve similar results by incorporating reversals with the simpler basic momentum method.

Finally, and most importantly, is the challenge of explicability. Although an LSTM's architecture is simple to understand and implement, it is challenging to explain its performance which would not be acceptable for it to be deployed in a proper investment management setting.

References

Alpaydin, Ethem. 2014. *Introduction to Machine Learning.* London: MIT Press.

Cici, Gjergji, and Rabih Moussawi. 2004. *Momentum Strategies.* Nov. https://wrds-www.wharton.upenn.edu/pages/support/applications/portfolio-construction-and-market-anomalies/replicating-momentum-strategies-jegadeesh-and-titman-jf-1993/#references.

DeMiguel, Victor, Lorenzo Garlappi, and Raman Uppal. 2009. "Optimal Versus Naive Diversification: How Inefficient is the 1/N Portfolio Strategy?" *The Review of Financial Studies* 1915-1953.

Dreschler, Song Qingyi. 2018. *Momentum Strategies (Python).* May. https://wrds-www.wharton.upenn.edu/pages/support/applications/portfolio-construction-and-market-anomalies/replicating-momentum-strategies-jegadeesh-and-titman-jf-1993-python/.

Eisdorfer, Assaf . 2008. "Delisted firms and momentum profits." *Journal of Financial Markets* 160–179.

Fama, Eugene. 1970. "Efficient Capital Markets: A Review of Theory and Empirical Work." *The Journal of Finance* 383-417.

Fischer, Thomas G. 2018. *Reinforcement learning in financial markets - a survey.* Econstor.

Fischer, Thomas, and Christopher Krauss. 2018. "Deep learning with long short-term memory networks for financial market predictions." *European Journal of Operational Research* 654-669.

Jegadeesh, Narasimhan, and Sheridan Titman. 2001. "Profitability of Momentum Strategies: An Evaluation of Alternative Explanations." *The Journal of Finance* 699-720.

Jegadeesh, Narasimhan, and Sheridan Titman. 1993. "Returns to Buying Winners and Selling Losers: Implications for Stock Market Efficiency." *The Journal of Finance* 65-91.

Krauss, Christopher, Xuan Anh Do, and Nicolas Huck. 2017. "Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P 500." *European Journal of Operational Research* 689-702.

Lim, Bryan, Stefan Zohren, and Stephen Roberts. 2019. *Enhancing Time Series Momentum Strategies using Deep Neural Networks.* SSRN.

Medsker, L.R., and L.C. Jain. 2001. *Recurrent neural networks: Design and applications.* New York: CRC Press.

Olah, Christopher. 2017. *Understanding LSTM Networks.* August 25. Accessed July 16, 2019. http://colah.github.io/posts/2015-08-Understanding-LSTMs/.

Takeuchi, Lawrence, and Yu-Ying Lee. 2013. *Applying Deep Learning to Enhance.* Stanford University.