Online non-linear prediction of financial time-series patterns

Mr Joel da Costa
Supervisor: Prof. Tim Gebbie

University of Cape Town
MSc. Advanced Analytics (STA5004W) Research Proposal

Abstract

We consider pattern prediction of financial time-series data. The algorithm framework and workflow is developed and proved on daily sampled OHLCV (open-high-low-close-volume) time-series data for JSE equity markets. The input patterns are based on pre-processing using FFT data interpolation for missing and outlier data points. The input data vectors are equal size data windows pre-processed into a sequence of daily, weekly and monthly sampled feature measurement changes (here log feature fluctuations). The data processing is split into at offline batch processed step where data is compressed using an autoencoder via unsupervised learning, and then batch supervised learning is carried out using the data-compression algorithm with the output being a pattern sequence of measured time-series feature fluctuations (log differenced data) in the future (ex-post) from the training and validation data. The historical simulation is then run using an online neutral network initialised with the weights from the offline training and validation step. The historical simulation is then considered in terms of test for statistical arbitrage and a simple correction for transaction costs is considered.

Keywords: online learning, neural network, pattern prediction, JSE, non-linear, financial time series

1 Hypotheses

- Primary Hypothesis: A online non-linear model can be trained to predict pattern matches in OHLCV securities data
- Secondary Hypothesis: Preprocessing of data through an autoencoder can be shown to produce statistically significant improvements

2 Literature Review

Technical analysis is a financial analysis practice that makes use of past price data in order to identify market structures, as well as forecast future price movements. The techniques are typically objective methodologies which rely solely on past market data (price and volume). They stand in contrast to fundamental analysis, where experts will consider a companies operations, management and future prospects in order to arrive at an evaluation. The basis of much technical analysis, originally developed through Dow Theory, is the belief that stock market prices will move directionally (upwards, downwards or sideways), and that past movements can be used to determine these trends [1].

One of the primary methods in technical analysis is the use of charts in order to identify price patterns. These charts will be produced using the available market data and a known design, such as the popular candle-bar plot, which can then be compared to historical data to match it to a particular pattern. These patterns are thus indicative that the stock is likely to take on a particular price trend, or is in a particular state [1]. There is a certain amount of controversy around technical analysis, where many argue that it is contradictory to the random walk and weak form efficient market hypotheses, and as such is not valuable or useful [2]. The argument against this, is that technical analysis does not rely on past action to predict the the future, but is rather a measure of current trading, and how the market has reacted after similar patterns have occurred in the past [3]. Further, even if the analysis is unable to effectively forecast future price trends, it can still be useful to exploit trading opportunities in the market [4].

With the advent of processing power becoming cheaply available, there has been an increase in research to adapt computing techniques to technical analysis. The breadth and superhuman speed in which systems are able to perform technical analysis far outstrips what was possible before, and as such they have become the focus of competitive performance for many market participants [5]. To this end, there has been much research to apply machine learning algorithms to perform pattern recognition on stock price movements.

Financial markets have been shown to be complex and adaptive systems, where the effects of interaction between participants can be highly non-linear [6]. Complex and dynamic systems such as these may often exist at the 'order-disorder border' - they will generate certain non-random patterned and internal organisation, which can be assessed and identified, however they will also exhibit

a certain amount of randomness in their behaviours, or 'chaos' [7]. As a result, trying to identify these patterns and structures is a simultaneously reasonable and notoriously difficult goal. While it is often clear in hindsight that the patterns exist, the amount of noise and nonlinearity in the system can make prediction challenging.

Fittingly then, neural networks have become a popular choice for modelling within the financial markets. Due to their structure, they are able to learn non-linear interactions between their inputs and outputs, with even early research showing their ability to achieve statistically significant results [8]. They have been shown to be universal function approximators: given the correct data and internal structure, they are able to model any input and output relationship [9]. Neural networks have been shown to be capable of generating trading signals which outperform traditional buy and hold strategies - simultaneously refuting the efficient market hypothesis, as well as demonstrating the networks capacity to capture the non-linear relationships of the market [8].

More recent work has lead to the development of convolutional neural networks, and more generally, deep belief networks. These take the traditional feed forward networks with single hidden layers, and extend both the layered architecture, as well as the processing between layers. Research such as ImageNet has shown this structure to be incredibly adept at pattern recognition [10][11]. This structure has been further adapted to show effectiveness at predicting time series data which outperforms base neural models [12].

Pang et al. have shown a novel approach in deep belief networks layering auto encoders prior to the neural networks (long short-term memory in this case) which can be effective, though the results leave room for improvements [13]. Bao et al. also showed that the use of stacked auto encoders in a deep belief network (LTSM again) lead to better performance than other neural network model without the auto encoders [14]. On a similar note, Hatami et al. were able to show that converting one dimensional time series data to a 2 dimensional recurrence plot was an effective way in improving prediction results for convolutional neural networks [15]. Evidently, there is room and reason for investigation in how preprocessing of time series data is able to increase performance in neural networks.

There is a certain disconnect in some of the approaches mentioned here and the practicality of implementing them in a financial context. Many machine learning and predictive models require extensive batch training, with the assumption that there is an substantial or full range of input covered. In reality, particularly in the realm of financial prediction, input is only partially available and further input will continue to arrive in the future. Models that are able to operate by updating themselves once more data becomes available are known as 'online' models [16]. These are naturally applicable in the realm of financial time series data. Some of the more extensive deep belief models, such as LTSM, can have time consuming training times [14], which may negate their use for an online network.

Suggested financial strategies are typically justified through backtests - a simulation of the investment strategy on historical data. In these simulations, the strategies performance over a set of data is recorded to determine the profits and losses it would have generated, as well as the more popular financial indicators such as the Sharpe ratio (used to quantify the return on risk). There is an important distinction here between the In-Sample (IS) and Out of Sample (OOS) datasets. Ideally, the algorithm is developed on the training, or IS data, and tested on the unseen validation data, or OOS data. Bailey et al. [17] have discussed and shown that both over fitting and data snooping are rife in the literature on trading strategies. These issues within the backtesting simulations largely invalidate the results for practical purposes, as theres no clear evidence of how the strategy would operate in the market. There is a clear parallel in the structuring of data for backtests and for training machine learning models. However, as Bailey et al. have pointed out, the methodology is not always applicable. There are difference in outputs (e.g. point forecasts versus trading signals), potential problems with overfitting other aspects of a trading strategy (e.g. entry or stop-loss thresholds), and a lack of accounting for the number of trials attempted in regression fitting. Bailey et al. later developed an effective and novel method (combinatorially symmetric cross-validation) of running backtest simulations in such a way that overfitting is unlikely to occur, as well as offering a probabilistic measure of whether it has occured or not [18].

With consideration to the aspects raised above, this project will aim to implement an online, feed-forward neural network to recognise patterns in financial time series data, and so predict feature fluctuations. Feedforward networks will be used so as to make the online updating viable in terms of speed. Consideration will be given to the preprocessing of data and using auto-encoders in order to improve performance. The model will then be trained and tested, using appropriate backtesting methodolgies (as per [18]), on OHLCV data from the JSE, with expansion to intraday data afterwards.

3 Aims and Objectives

1 Create a feature set artificially made up of time-series data slices with a bespoke convolution setup
   - This will require data pre-processing that is fast and which can be both online and offline
   - FFT Interpolation will be used for missing data

2 Compare daily sampled data from various difference equity classes

3 Use dimensional reduction through an auto encoder to show performance improvements

4 Create a library which is able to generate the full online non-linear prediction model through the use

of auto-encode and an online neural network. This needs to be able to achieve n-order prediction both online and offline

  - The typical basis of statistical learning will need to be considered: training, validation and testing

5 Create backtesting module for the online model

  - This will be novel if it is correctly implemented online step t to t+1 without any batch update and correctly accounts for cost

6 Discuss the generalisation error, as well as in-sample vs out-sample errors and performance statistics of the backtesting results

7 Consider the use of the online model for a profitable trading strategy

  - Trading and transaction costs will need to be taken into account
  - Carry out a standard hypotheses test for a statistical arbitrage (test for positive probability of excess returns, vanishing variance and zero probability of gamblers ruin.)

4 Data Requirements Specification

- A surrogate dataset of one or two stocks will be used to start development
- Thereafter a fuller set of data will need to be collected, from Bloomberg and Thomson-Reuters
- Bloomgberg OHLCV daily data for 20 years will be considered in the stock combinations as detailed in Table 1
- Thomson Reuters Tick History intraday data consisting of top-of-book and transaction updates for the same stocks as listed in Table 1
  - Data will be processed to create 5 minute and 10 minute bars from the intraday data as well as volume time bars to be used as an input to the online learning algorithm
- Synthetic data cases (Eg. Monte Carlo simulations) will also be considered in order to discuss issues encountered with in-sample versus out-of-sample backtesting
  - Examples of such data cases would be where stocks are all increasing/decreasing over time, or both for a combinations of stocks.
- As detailed in table 1, there are 5 primary stocks, each of which will be considered in various Stock, Equity Index and Bond Index pairs, as well as by themselves. E.g. For AGL, the following will be considered: AGL, AGL and BHP, AGL and ALSI40 and AGL and ALBI
- The Daily TRI (Total Return Index) OHLCV for 20 years will be considered for all pairs, and the Intraday data will be considered for the Single and Stock pairs.)

Stock	Stock Pair	Equity Index	Bond Index
AGL	BHP	ALSI40	ALBI
SBK	SNL	ALSI40	ALBI
SHF	RCH	ALSI40	ALBI
WHL	SHP	ALSI40	ALBI
MTN	VOD	ALSI40	ALBI

Tab. 1: Stock Combinations

5 Data Science Workflow

The project will be split up into several stages, as per the processes that the data will go through. These are detailed below, and expanded on in [19]

1 FFT Interpolation: This will process the raw OHLCV data, and inerpolate any missing points using FFT

2 Data Processing: Using the now complete dataset, this will layer multiple time slices of the data's feature fluctuations

3 Dimension Reduction: This layer will perform dimension reduction on the time sliced data using an Autoencoder

4 Offline Neural Network: A standard ANN using data processed through the Autoencoder as input to predict n-step fluctuations

5 Online Neural Network & Backtesting module: An online version of the offline neural network, validated using testing techniques similar to those detailed by [17] [18]

6 System Requirements Specification

- Hardware: A macbook pro will be use for the crux of the development, with the following specs: 2,8 GHz Intel Core i7, 16 GB 2133 MHz LPDDR3, SSD
- Software: Julia will be the primary programming language used to develop the project, though Python and R will also be used interchangeably for Exploratory Data Analysis (EDA), and as needed. There may be

various public libraries used within all of these languages that are considered at the time

- Each of the project stage deliverables will be done by following the process detailed in the steps below. These are detailed further in [19]

  1 Problem Definition

  2 Data Processing

  3 Data Exploration

  4 Baseline Modeling

  5 Secondary Modeling

  6 Testing

  7 Process Preparation

7 Project Milestone Deliverables

Date	Description	Deliverable
31/03	Literature Review & Learn Julia Basics	
21/04	FFT Data Interpolation	FFT Library
15/05	Synthetic Data Generation	Synthetic Data Collection
31/05	Data Preprocessing	Processing Library
21/06	Dimensional Reduction	Auto Encoder class
10/07	Finalise Data Collection	Final Dataset
05/08	Offline Neural Network	Training Library, Model, Validation Test Outputs
31/08	Online Neural Network	Process Library and Model
30/09	Backtesting Module	Process Library and Statistical Test Results
31/10	Testing on Exteded Datasets	Models and Statistical Test Results
30/11	Dissertation Write Up and Revisions	Final Hand-in

Tab. 2: Key dates and deliverables for research project

8 References

[1] John J. Murphy, Technical Analysis of the Financial Markets (New York Institute of Finance, 1999), pages 1-5,24-31.

[2] Griffioen, Gerwin A. W., Technical Analysis in Financial Markets (March 3, 2003).

[3] Kahn, Michael N. . Technical Analysis Plain and Simple: Charting the Markets in Your Language, Financial Times Press, Upper Saddle River, New Jersey, p. 9. ISBN 0-13-134597-4.(2006)

[4] Jack D. Schwager. Getting Started in Technical Analysis, Page 2 (1999)

[5] N. Johnson, G. Zhao, E. Hunsader, H. Qi, N. Johnson, J. Meng Brian Tivnan (2013). Abrupt rise of new machine ecology beyond human response time. Sceintific Reports 3(2627). DOI: 10.1038/srep02627

[6] B. Arthur (1995) Complexity in Economics and Financial Markets, Complexity 1 (1), 2025.

[7] J. P. Crutchfield, (2011), Between order and chaos, Nature Physics, Vol. 8, January 2012, 17-23

[8] Skabar, Cloete, Networks, Financial Trading and the Efficient Markets Hypothesis (http://crpit.com/confpapers/CRPITV4Skabar.pdf)

[9] K. Hornik, Multilayer feed-forward networks are universal approximators, Neural Networks, vol 2, 1989

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, ImageNet Classification with Deep Convolutional Neural Networks, Advances in Neural Information Processing Systems 25, (2012), pp. 10971105.

[11] Aaron van den Oord, WaveNet: A Generative Model For Raw Audio (19 September 2016) `https://arxiv.org/pdf/1609.03499.pdf`

[12] A. Borovykh, Conditional Time Series Forecasting with Convolutional Neural Networks (Oct 18th 2017) `https://arxiv.org/pdf/1703.04691.pdf`

[13] X. Pang, An innovative neural network approach for stock market prediction (12th Jan 2018) `https://link.springer.com/article/10.1007/s11227-017-2228-y`

[14] W. Bao, A deep learning framework for financial time series using stacked autoencoders and long-short term memory (14 July 2017) `http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0180944`

[15] Nima Hatami, Classification of Time-Series Images Using Deep Convolutional Neural Networks (7th Oct 2017) `https://arxiv.org/pdf/1710.00886.pdf`

[16] S. Albers, Ser. B (2003). Online algorithms: a survey. Mathematical Programming 97(1). doi:10.1007/s10107-003-0436-0

[17] D. H. Bailey, J. M. Borwein, M. M. Lopez de Prado, Q. J. Zhu (2014). Pseudo-Mathematics and Financial Charlatanism: The Effects of Backtest Overfitting on Out-of-Sample Performance. Notices of the AMS, 61(5), 458-471

[18] D. H. Bailey, J. M. Borwein, M. M. Lopez de Prado, Q. J. Zhu (2015). The Probability of Backtest Overfitting

[19] J da Costa, 'Online non-linear prediction of financial time-series patterns, Technical document accompaniment (2018).