Effect of Algorithmic Variables in LSTM's Prediction of Constituent Direction of the S&P 500: as Applied to the EMH

AP Research

May 2, 2018

Word Count: 4818

Optimizing the LSTM: as Applied to the EMH

Introduction

Since the inception of applying computational techniques to the economic sector in the 1950s, the demand for accurate prediction of volatile, nonlinear systems has driven the creation of mathematical models aimed at market forecasting and optimization. Critics of this field of technical analysis (TA), a discipline of evaluating investments and opportunities based on trends and historical signals, propose the market is random and therefore is unpredictable. Conversely, practitioners of TA assert that the market must not be random since because researchers who aim computational resources at financial prediction through data-driven machine learning (ML) approaches have accurately predicted market trends (Marr, 2016). Although the literature is unresolved, opposition to TA often stems from applying William Sharpe's Efficient Market Hypothesis (EMH). The EMH's principal postulate is that markets are efficient, as they already best reflect all available information for each constituent and therefore the current prices reflect all past prices (Sharpe, 1966).

Implications of this theory are evident — predictive signals are already represented to investors through current information within the stock price, and unless new financial information makes its way into the fiscal limelight, there is no under or overvalued equity. This perfect representation creates unpredictability as past performance or trends have no further effect on future directions in an efficient market. This paper aims to address what findings TA can offer to the EMH and explains the degree to which markets are efficient, the causes of their inefficiencies, and the current impact of ML on efficiency. Additionally, this paper examines algorithmic variables within the Long Short-Term Memory network

(LSTM), a recurrent ML algorithm, that optimize the program for nonlinear time-series prediction.

Literature Review

Machine Learning is the process of computational learning. The primary aim is to allow computers to learn automatically without human intervention or assistance and adjust their actions accordingly. As first identified by Bayes (1763) and Turing (1950), ML algorithms all start with observations, or data, and begin looking for patterns within the data to make better decisions in the future. These algorithms extrapolate from what was learned from provided examples and apply it to new data. In this paper's method section, the mathematical functions and processes of ML will be examined more closely.

The history of utilizing machine learning to predict and identify the nonlinear dependencies in stock markets dates back to 1952, when Nobel Prize winner and UCSD professor of finance Harry Markowitz first addressed the implementation of computers in finance by treating portfolio selection as a simple mean-variance optimization problem (Markowitz, 1952). Finding the computational power of the available machines at the time insufficient, his work shifted to simple relationships that would not require vast amounts of sophisticated decimal calculations to evaluate (Fox & Sklar, 2009). During the 1960s, as computational power increased, Markowitz and other researchers worked with hedge fund managers to pioneer the use of statistical arbitrage, a computational approach to algorithmic commodity trading (Goodkin 2012). Although pioneers like Markowitz proposed more advanced forecasting algorithms at that time, it was not until the 1990s that ML was reliably applied to financial prediction, mostly due to the availability of well-maintained, reliable data and the exponential increase of available computational power and resources (Fox & Sklar, 2009). Researchers soon began implementing new techniques such as sentiment analysis, signal processing, and time-series forecasting to financial problems (Ehlers, 2001). The trailblazers of TA vastly expanded the ML research base as well as the academic EMH discussion, and their work continues to inspire new waves of research today.

Currently, three main classes of researchers conduct the body of technical analysis research. The first class contains academic economists like Jacobs (2015) who contribute to identifying what signals to process, input data to use, and what information-based trading strategies are profitable. Although these groups excel at addressing the financial perspective, they often lack the computational intuition to know how to best optimize their model (intelligent piece of software), to best format their inputs, or to best select an algorithm. On the other side of public academia is a class of computer scientists like Porshnev, Redkin, and Shevchenko (2013) who also make notable contributions to the available literature. These researchers have excellent intuition regarding what algorithms to use, how to best train them, and how to optimize them, but lack the insight into trading strategies, market anomalies to track, or what data best represents a company's performance. These academic discrepancies cause a rift between academic research and private practice. The third class is the private sector. As Krauss, Do, and Huck (2016) describe, private companies successfully employ a variety of ML strategies to annualize high returns, and combine the skills of employees with backgrounds in both academic disciplines. Unfortunately for public researchers, the need for corporate secrecy prevents the best and most effective methods from being published, keeping them under industry guard. This furthers the wedge between what is known to be possible, such as the 98.2% returns for the 2008 fiscal year Rubin and Collins (2015) describe, and what has been reproduced or even attempted. Since those in the private sector who may benefit from fresh research already know and practice what academics publish as novel, no truly groundbreaking work is produced.

Some of the more notable research that approaches financial time series forecasting while bridging a gap in the research base include Schumaker and Chen (2006), Krauss, Do, and Huck (2016), and Fischer and Krauss (2018). University of Arizona researchers Schumaker and Chen's (2006) Arizona Financial Text System (AZFinText) utilizes support vector machines (SVM) to compute sentiment and textual analysis on financial news documents. The program predicts near-term direction of non-bias-eliminated constituents of the S&P 500 and was able to improve returns of a simulated trading agent by a statistically significant amount but did not make statistically accurate predictions. These trailblazing professors set the tone for the academic research that followed but failed to reach across the academic aisle to create more solid trading strategies or market predictors.

Most early 2000s research was devoted to low-accuracy SVMs. As machine learning became more powerful and pervasive, researchers began comparing more modern algorithms and optimizing them. One such research group is Krauss, Do, and Huck (2016), from the University of Erlangen-Nürnberg and ICN Business School. Aiming their research at "bridging [the] gap" between academic finance and the financial industry, these researchers employ Deep Neural Network, Gradient Boosted Trees, and Random Forests, all advanced machine learning algorithms that work across distinct statistical and computational niches (Krauss, Do, & Huck, 2016, p. 2). A key finding from this study is the ability of Random Forests to produce statistically significant returns with a higher degree of accuracy (~55%) than that of the other two models for a smaller stock portfolio. Krauss, Do, and Huck (2016) successfully collaborated across disciplines to create more powerful models and developed deeper understandings of how the market acts through statistical arbitrage and technical analysis. Despite their efforts, the researchers still acknowledge the private sector's ability to outperform them. Chiefly, the researchers cite the private quantitative hedge fund Renaissance Technologies' 98.2% annualized returns in 2008 as being far greater than their simulated 73% (Rubin and Collins, 2015; Krauss, Do, & Huck, 2016). Interestingly enough, both Krauss, Do, and Huck (2016) and Renaissance Technologies find spikes in market inefficiency at times of high volatility and emotion-based trading, and therefore noting higher returns. Building further on this research, Fischer and Krauss (2018) compares the LSTM to the other models in Krauss, Do, and Huck (2016). Being one of the first researchers to examine how the 'unreasonable effectiveness' of the LSTM could be applied to financial series as well, Fischer and Krauss (2018) finds the LSTM to be the most accurate model, with averaged yearly accuracy approaching a statistically significant ~55%. These researchers provide the basic methods much of this paper follows.

While recent literature tries to bridge the gaps in public and private ML research and TA, there still exists a considerable amount of research to be done in the field of ML optimization. Like any mathematical algorithm, there are variables within each ML algorithm that change the effectiveness of the model. Stanford researchers Alice Zheng and

Jack Jin (n.d.) detail a seemingly infinite combination of algorithms and algorithmic variables exist when it comes to designing a ML project. The team outlines the following variables that could be examined throughout any given project:

- Forecast Horizon – The period in the future the model is set to predict. In the cases of stock forecasting, this could be on the microsecond, minute, hour, day, week, month, quarter, semester, year, decade, or any other arbitrary period.
- Batch Size – The size of the data and number of training examples used in one iteration. In the case of stock forecasting, this could be one company, one fund, one industry, one market, several markets, or any other arbitrary grouping of equities.
- Features – The measurable property of the phenomenon being observed. In the case of stock forecasting, this could be indicators such as a company's open price, close price, volume, high, low, market capital, return indices, yield, percentage change, or credit ratings. This could also be different types of input data such as global news, popular sentiment, company filings, or any other performance metric or bit of company information.
- Targets – The targeted measurable characteristic being predicted. In the case of stock forecasting, this could a be company's open price, close price, asking price, offering price, volume, high, low, market capital, return indices, percentage change, or direction for any timestep in the future.
- Machine Learning Algorithms – Mathematical approaches to predicting targets from inputted features with over 250 unique algorithms. Notable classes of algorithms include decision tree algorithms, Bayesian algorithms, artificial neural network algorithms, deep learning algorithms, recurrent neural networks, and more. Long short-term memory networks are specific examples of recurrent neural networks. Through ensemble learning, any number of these algorithms can be combined.
- Algorithmic Variables – The changeable values and equations within each algorithm that affect the effectiveness of the algorithm on a per-case basis. Such variables include the layer layout, model design, activation functions, loss functions, optimization functions, callback functions, size of the hidden layer, existence of dropout, and thousands of other variables as well.

All these variables can be mixed and compared in a seemingly-infinite number of ways, and each comparison makes for its own valuable research. Herein lies another issue with the current literature; it can be extremely wide in scope. It is somewhat of a fool's undertaking to test all possible models against each other and optimize each model. This daunting gap in the research creates a feedback loop where no individual or body tries to publish model optimization research on specific real-life datasets, causing entire fields to be left seemingly untouched. This fear of real-life datasets acts as a barrier to publishing and unnecessarily narrows the field. In their desire to be published, failed optimization efforts are cut from the literature and go unpublished, leading researchers to repeat inefficient methods. Not only does this widen the gap between research in the public and private sectors by slowing the pace of the public research, but it also discourages general ML optimization research which could have broad implications for other fields.

Areas of Inquiry

Because of the potential for an overreaching research scope, the methods defined take considerable care to narrow the focus to one metric, one algorithm, one type of input data, one index, a few algorithmic variables, and a well-defined, easily comparable baseline. This work asks, *How does the accuracy of a Long Short-Term Memory Network's prediction of the direction of the return indices of constituents of the S&P 500 vary when algorithmic variables such as the forecast horizon, the size of the hidden layer, and the existence of a dropout layer, are changed compared to the baseline of the fiducial LSTM, and what can be learned about the Efficient Market Hypothesis from the model's outputs?*

While the narrow scope limits the degree to which the model can be optimized, it should not prevent novel insights on the EMH from being derived. Because of the limited research base and lack of research into model optimization, this paper aims to contribute three main points to the contemporary discussion:

First, this paper offers a clear, replicable standard for a fiducial LSTM that future researchers could use for testing LSTM optimization. It also contributes findings on how to best optimize the LSTM to fit a specific type of time-series data, addressing what failed in testing. Second, this paper adds to a field lacking published literature, offering encouragement to explore model optimization on authentic datasets. It also largely follows the methods of researchers who worked across disciplines, preventing unhelpful or misguided research. Last, this paper reaches across the oppositional perspectives of technical analysis and the Efficient Market Hypothesis to use one in part to support the other, simultaneously broadening the understanding of both.

Methods of Processes of Algorithms

As previously discussed, ML algorithms start with a set of training data and begin looking for patterns within that data. They then extrapolate those 'learned' patterns to make decisions or predictions when confronted with unseen data. To do this, the algorithms run the data through a series of computations, returning predictions that, one hopes, will approach the intended goal over time.

The core unit of a neural network, a broader term for algorithms which model their prediction process in a manner similar to the human brain, is a neuron, or a node. Nodes can represent input, output, or some other hidden function. Groupings of nodes make up layers whose names are defined by what they do. The name for the layer comprised of nodes containing the input data is known as the input layer, and likewise for the output layer. Layers in which hidden computations map the input to the output and discover dependencies between the two are known as the hidden layers (*Figure 1*). [1]

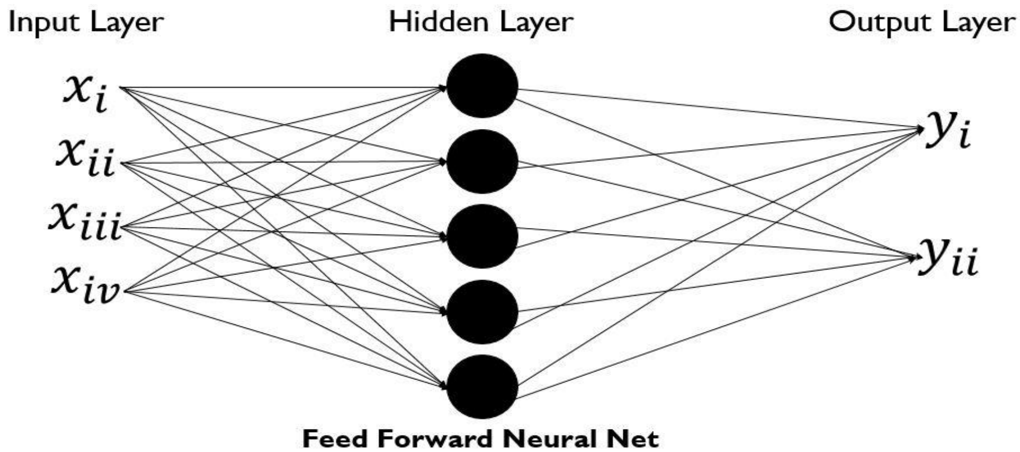

Figure 1: Each input layer element (x_n) maps to each hidden node, which maps to each output node (y_n) in a simple feed forward neural network. These models are used for recognizing simple linear dependencies, but often fail examining more complex systems.

Each element from the input layer (x_n), maps to each node in the hidden layer with its own unique weight. Then, each of these nodes in the hidden layer maps to its respective output (y_n). The shape of the output layer defines the returned targets. For example, a neural net with an output shape of two ($n=1$) will return a binary sequence, i.e. either a zero or a one. This defines a binary classification model that aims at predicting trends that are either active or are not active, usually denoted by positive or negative inputs. In this paper, the binary return equates to the direction of the constituents, i.e. if the equity will trend upward *(1)* or downward *(0)* over the course of the day, if the close is greater than or less than the open.

[1] In order to better help the reader understand of the processes of machine learning, the researcher has created several original visual aides to illustrate the complicated processes more clearly.

Examining the process more closely (*Figure 2)*, each weight from the input layer (w_n) is summed in each hidden node it maps into, creating a bias vector ($\overrightarrow{w_n}$). This vector is then passed through an activation function, usually a Sigmoid, Tanh, or SoftMax function, acting as a stepper-function that determines if the neuron will 'fire' or not, like the human brain. If the summed weight is greater than a certain threshold, the neuron will 'fire' and return. Modern activation functions use floating-point values denoting how certain it is of the answer rather than zeroes and ones.

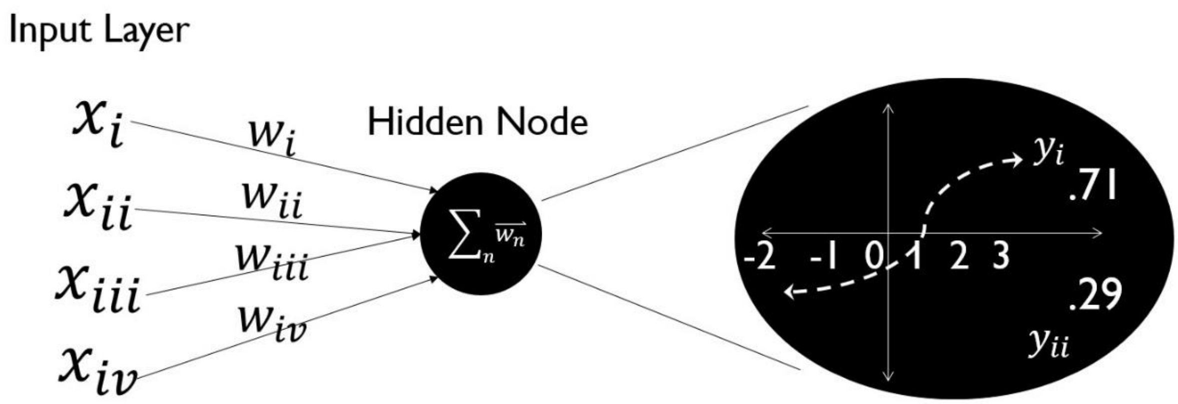

Figure 2: Each element of the hidden layer (x_n) maps to a node in the hidden layer with a unique weight (w_n). Based on the sum of the weights ($\sum_n \overrightarrow{w_n}$), the neuron will either return or not. In this case, the node would be 71% certain of its answer if $\overrightarrow{w_n} \cong 1.7$, and would return $> .5$, the mathematical equivalent of a 1, firing the neuron. Note: the numbers expressed in the figure are trivial and are provided as an example.

These simple feed forward neural networks work well when the outputs are solely based on the previous inputs, but they fail to connect previous data with the present task. Thus, for targets in which historical trends play a role in the output, this algorithm falls short in accurate predictions. A popular solution to this issue is implementing a Recurrent Neural Network (RNN) (*Figure 3)*, a specific type of neural network that creates mathematical memory within the algorithm that can be used to solve issues that are associated with short-term time-series predictions. It does this by not only passing through input data into the hidden layer, but also the state of the previous hidden layer to create a layered view of the data that includes previous biases, memories.

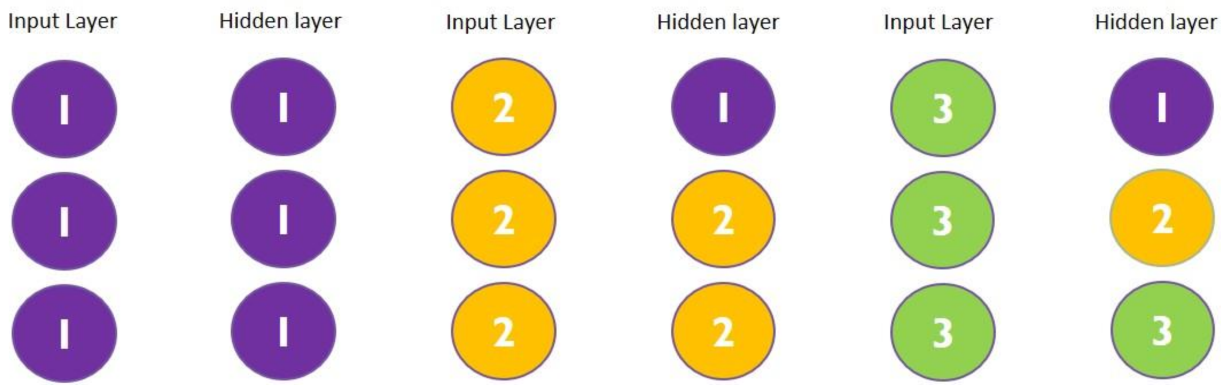

Figure 3: Previous memories are represented in future timesteps as the state of the previous hidden layer connects past sequences with the present through more than just the pre-established biases.

Unfortunately, over time, this solution becomes problematic. As identified in Yoshua Bengio's frequently-referenced 1994 paper, "Learning long-term dependencies with gradient descent is difficult" in the IEEE Transactions on Neural Networks, in simple RNNs, memories become subtler as they phase into the past. Error signals from previous timesteps do not make it far enough through each epoch, or iteration over the training data, to make any effect in timesteps far down the line, regardless of how strong the signal is. This is because the weights of the signals are continually updated through vectored decimal multiplication, making their value smaller each epoch, creating a vanishing gradient (*Figure 4).* Understanding the problems to a model posed by the threat of a vanishing learning gradient reveals the ways in which the selected LSTM model combats them.

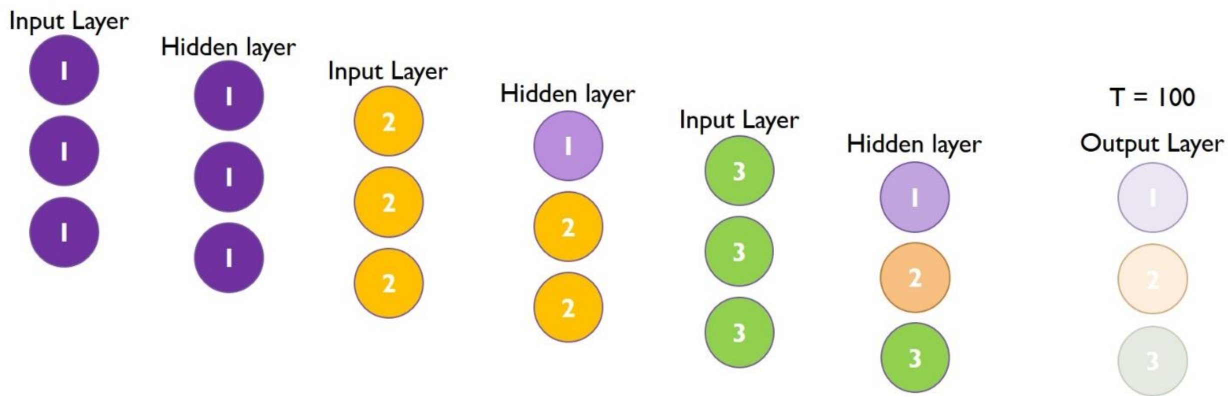

Figure 4: Memories from previous timesteps become less pronounced as they phase into the past. By arbitrary timestep 100, the outputted memories from the first few timesteps are so dim they barely affect any decision, if at all. This makes learning long-term dependencies difficult with a simple RNN.

As first described by Hochreiter and Schmidhuber (1997), an effective method to address the vanishing gradient is a specialized RNN known as a long short-term memory network (LSTM). An LSTM maintains its errors signals by replacing ordinary neurons in the hidden layer with specialized memory cells. Memory cells have an internal feedback loop multiplying all bias by one, as any number multiplied by one equals the original number. This preserves error signals indefinitely in the model, making short-term memories permeate past their expected vanish point, hence long short-term memory. To avoid preserving all memories, the memory cell has forget gates at each input, and each gate has its own activation function that learns when to leak error as well as when to filter it, preventing extraneous memories from forming.

This paper choses to examine the LSTM for a few reasons. First, the LSTM already displays a well-documented ability to forecast time series to a high degree of accuracy, and its internal gates allow for an easy filtration of statistical noise, something financial data is notorious for having in high degrees. Additionally, the ability to handle vanishing gradient problems allows the model to contextualize the market in ways other models fail. Finally, through Fischer and Krauss (2018), a well-documented model of an LSTM has already been benchmarked for these methods to reproduce and expand on, as well as establish as the fiducial model for easy comparison.

This paper also identifies three algorithmic variables to optimize over: (1) the role of the size of the hidden layer, defined as the number of stacked layers of five memory cells; (2) the forecast horizon, or how far into the future the model is to predict; and (3) the existence of dropout in an LSTM's prediction. Dropout is a mathematical technique that randomly drops data from the training set, making the data different each time to prevent the model from overfitting, or memorizing, the training data instead of learning the dependencies between inputs and outputs. Overfitting makes predictive models that cannot perform well when challenged with new data, defeating the ML element as no true learning occurs.

Methods of Research Design

To gather training data, all current and historical constituents, or member stocks, of the S&P 500 since January 1, 1997 to January 1, 2017 were compounded using Thomson Reuters Eikon's Leavers and Joiners function (Thomson Reuters Eikon, 2018b).[2] Over that period, the historical daily return index for each constituent was returned using a series of

[2] Since after the use of the software but before the finishing of this paper, Eikon has changed hands to Refinitiv, but remains cited as Reuters' tool as that is the version worked with, generously provided by the Interdisciplinary Research Collaboratory at UCSB (See McDowell 2018).

call functions (Thomson Reuters Eikon, 2018a). The data was cleaned and compounded in using Python 3.7, a programming language with several packages that make ML programming an easier task. Next, for all trading days in which any constituent was not represented on the index, the values of all return indices were replaced with the floating-point value 'inf' to avoid computation or learning on these values. This forces the model to only train on days represented on the index, and therefore erases as much bias in the model as possible from the inputs. The data was normalized using studentized residual normalization *(Eq. 1).* The data was then split into testing and training groups, following a 35:65 ratio.

$$\frac{\hat{\epsilon}_i}{\hat{\sigma}_i} = \frac{X_i - \hat{\mu}_i}{\hat{\sigma}_i}$$

$\hat{\epsilon}_i$ = Return index for time period i
X_i = Single day return at day i
$\hat{\mu}_i$ = Standardized mean of all data on index at time period i

Equation 1

Normalizing creates more accurate and efficient models by eliminating much of the noise while keeping the core dependencies between the values in any given set. Return indices are cumulative-divided prices that account for all relevant corporate actions, splits and inflation, making them the natural choice as the best metric of a company's performance. The S&P is the largest and most representative index of the U.S. economy, with less self-correction than the DOW Jones Industrial Average. Training the model on all current and historical S&P constituents also eliminates survivor bias in the forecasts. If the model were only trained on stocks which had not been dropped from the index due to poor performance, the model would only learn how to predict successful stocks, already a statistical anomaly.

Model Design

For reproducibility and baselining purposes, the fiducial LSTM architecture follows Fischer and Krauss (2018). The layout of the model is as follows:

- A dense input layer of shape 240 timesteps (one trading year) and one feature (one company at a time)
- An LSTM hidden layer of 25 neurons with a dropout of dr = 0.01
- A Dense output layer of 2 neurons, defining a binary classification problem, with a SoftMax activation function
- The model was compiled and optimized using RMSprop (lr = 0.001) with a binary crossentropy loss function
- The model was evaluated on metrics for accuracy and loss rate
- To avoid overfitting or unlearning, several callback functions were used to terminate the process and save the model if negative learning rates were encountered
- Tensorboard was used for data visualization (See Abadi et al. (2016))

All code was compiled and run in Python 3.7 using Google Colab, a powerful cloud-based computing resource for machine learning education and research built on the Jupyter notebook framework. Notable packages utilized include Keras, a ML library built on the Tensorflow framework, Pandas, a package for data manipulation, MatPlotLib, a library for data visualization, and NumPy, a package for advanced mathematics and data manipulation (Chollet, 2018; Abadi et al., 2016; McKinney, 2010; Hunter, 2007; Oliphant, 2006). Because of the utilization of Google Colab, all models were trained on either server-grade Nvidia Tesla K80 GPUs, server-grade Nvidia Tesla T4 GPUs, or, when available, Google cloud TPUv2s.

Results

First, the researcher evaluates how well the fiducial model responds to testing data. On average over the twenty-year period, the model was able to accurately forecast the direction of constituents 55% of the time. While 5% over random chance may not sound statistically significant to the reader, over 12 million datapoints were used, and the probability distribution of a random guessing agent (*Figure 5)* would suggest that the LSTM's results have less than a 1% chance of being reproduced by a random guessing algorithm. Showing the model is making intelligent predictions supports the assertion that machines can accurately forecast markets.

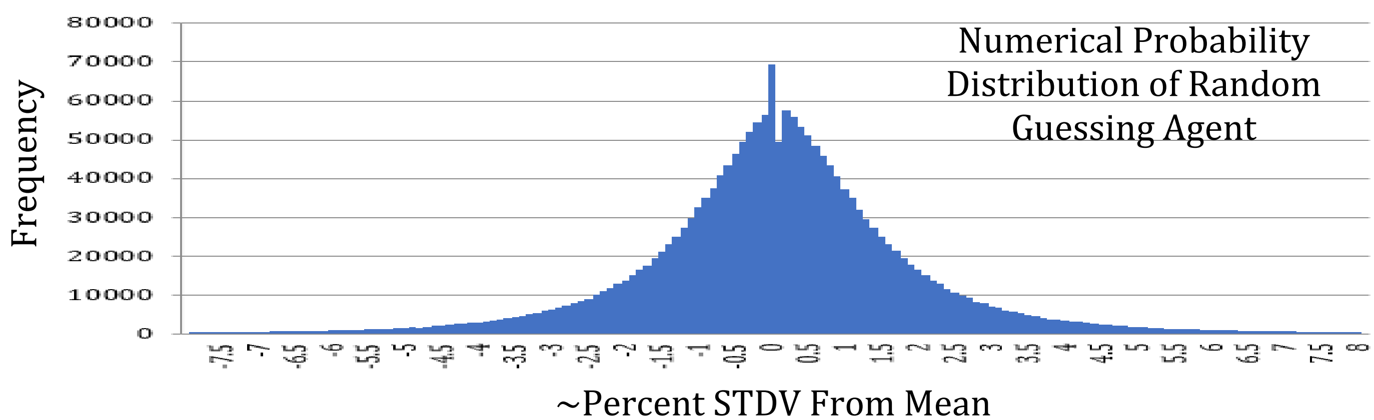

Figure 5: Following a Gaussian curve, the probability of a random guessing agent returning the direction of the return indices >51% of the time at a daily level is astronomically improbable.

Contrary to some of the findings of Krauss, Do, and Huck (2016), the model maintained more statistically significant predictions on shorter forecast horizons. Because stocks trend upwards on longer timesteps, such as the month or the year, (otherwise they are removed from the index), the ~60% accuracy documented on these timesteps was not statistically significant; An unintelligent guessing agent could see this ~60% accuracy on only random guesses of positive direction, meaning the model was not much more well-adapted than random chance in the long run. Inexplicably, near-term weekly predictions also fell short of other researchers' results. Krauss, Do, Huck (2016) found weekly returns as the most accurately predicted forecast horizon, and yet with a directional accuracy of a statistically insignificant ~51%, the results were unable to be replicated or verified. While the fiducial model only seemed to return significantly on shorter timesteps, all models did learn, contrary to Siami-Namini and Namin (2018), which can be seen through the decreasing loss functions (*Figure 6).*

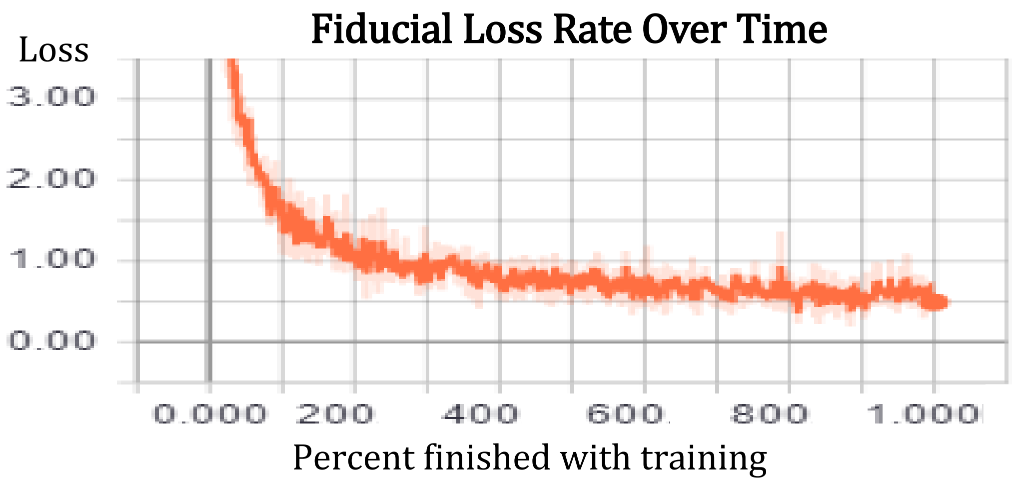

Figure 6: The loss function decreased as the model trained, showing the model successfully learning new dependencies between inputs and outputs – a promising sign in support of technical analysis' ability to accurately forecast nonlinear market systems.

In terms optimizing algorithms, the model with medium-sized 5-layer hidden layer (*n=5*) performs the best in comparison to the fiducial with an average annual accuracy of 57%. The fiducial model ranks second in terms of accuracy at 55% (*n=1*), followed by the least accurate model with a large 25-layer hidden layer at ~51% (*n=25*) (*Figure 7)*. The deficient performance of the n=25 model is likely due to a high degree of noise from competing logic gates in the hidden layer which provide too many conflicting error signals to drop and leak. Overall, the dropout technique was not necessary. A notable similarity between all the models these methods tested and previous models of Krauss, Do, and Huck

(2017), Fischer and Krauss (2018), and other researchers is their periods of best performance — periods of high volatility. During points in which the market was exhibiting substantial fluctuations and trades were emotion-based and speculative, such as the volatility surrounding the 2001 'dot com boom' or the 2008 mortgage and banking crisis, the models performed most accurately. Predictions from those periods were 15% more accurate than from periods of relative stability, or even healthy growth. Finally, the models replicated results from Krauss, Do, Huck (2017) in which their learning rates after 2001 decreased, which those authors attributed to a rise in autonomous ML market trading agents.

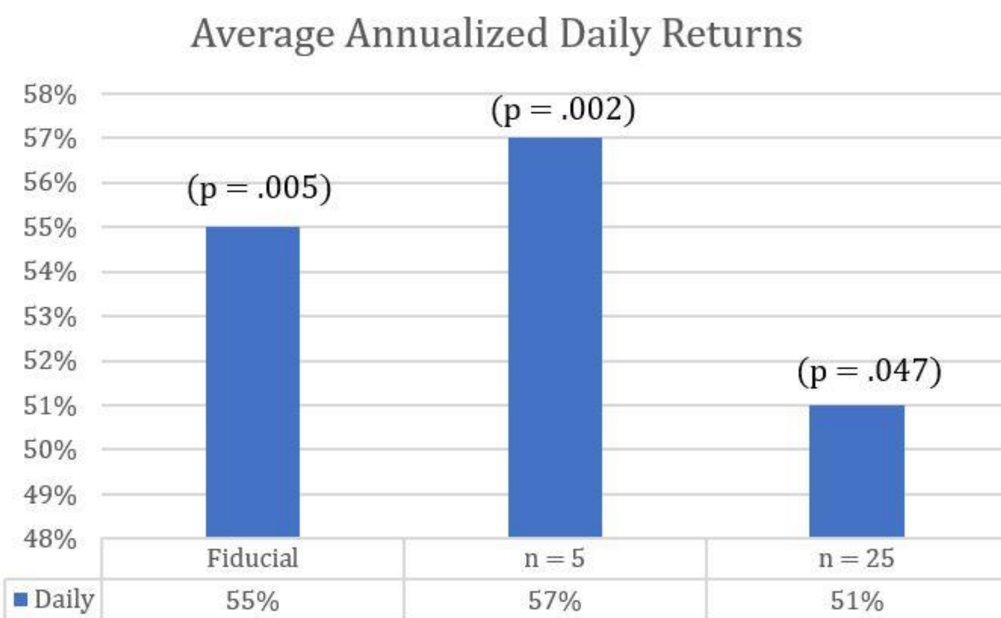

Figure 7

Conclusions on the Efficient Market Hypothesis

Valuable support for the EMH can be derived from the findings and trends the models identified, as well as the areas they underperformed. On shorter time scales such as the day, the market proves only semi-efficient, with a wealth of anomalies predicted by the best-fit model to a far higher degree of accuracy than random chance (57%). The ability for market direction to be predicted at all fundamentally disproves a pure EMH, but the fact that directional accuracy never approached some 90% still finds support for the market existing in a semi-strong form of efficiency during short timesteps. Since the market proves more efficient, and thereby less predictable, over mid to long timesteps, it seems market anomalies correct themselves quickly. While the market displayed some identifiable, shorter nonlinear dependencies, the market's general upward trend may have also accounted for much of the forecasting bias on longer timesteps, as the market moved efficiently to correct itself.

Some reasons for the why the market only displays semi-efficient characteristics on shorter forecast horizons supported through the results include the following (Green, Hand, and Zhang 2013; Fama, 1970):

- Stocks take time to respond to the latest information
- Human error, emotional trading, and undue speculation can affect stock prices

A pure EMH assumes that all stock prices would be instantaneously representative of all available information. However, as information diffuses to investors, there may be a brief period where new information is not evenly distributed, giving some investors a chance to exploit the inefficiency of a lagging market. Since the widespread implementation of ML and automated algorithm-based trading systems into the market in 2001, response time to new information has rapidly decreased as firms fight over network speed and latency rather than floor space. The marked reduction in learning rate across all models after 2001 was likely because of this change to automated systems with nearly instantaneous access to information. As the time to respond to new information is decreased by automated trading agents, the markets' efficiency has skyrocketed.

Similarly, markets are often least efficient when emotional trading or speculation is prevalent, supported by a spike in learning rates during periods of financial turmoil. However, through the implementation of ML and mathematics-based trading software into the market, an emotional and speculative element to market prices has largely been removed as over 75% of trades are made autonomously (Levine, 2013). Overall, this seems to stabilize the market while also making it less forecastable, and therefore riskier in some ways. Ironically, it is the machines' attempts to play to the market's inefficiencies that are simultaneously making the market more efficient, and therefore less-predictable.

Conclusions on Algorithmic Optimization

Undoubtedly, a LSTM with a medium hidden layer size predicting near-term daily direction produced the most accurate predictions annually when trained on American stock data. Dropout was unnecessary as the dataset was too large to overfit and the callback features made the technique a redundancy. It is important to note that because the model is limited to a narrow research scope by the small number of algorithmic variables examined, there still is vast room for development and optimization left unaddressed. Despite this, hypothetically, these findings on model optimization of the LSTM can also be applied to more diverse datasets, such as weather forecasting, disease spread predictions, or any other time-series prediction problem.

Suggested Further Work

There is still vast room for further inquiry since ML research is a mostly privatized, unpublished field that has an immense scope. As the models were limited, expanding only on some narrow variables, this paper could not fully follow through on a bulk of the algorithmic variables that could affect model performance. To fill this gap left in the research, future researchers could consider comparing further algorithmic variables or using several types of LSTMs (Bidirectional, convolutional, etc.) and comparing them to the fiducial model outlined in this paper. Additionally, researchers could identify batch size, additional layers, date format, or any other number of changes in comparison to the fiducial model. Other ML algorithms could also be tested, optimized, and compared. Training on foreign markets could also provide valuable insight into how different markets vary in efficiency. Finally, researchers could also evaluate the same model on more novel datasets or create trading strategies based on predicted directions.

Annotated Bibliography

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., ... & Kudlur, M. (2016, November). Tensorflow: a system for large-scale machine learning on heterogenous systems. In OSDI (Vol. 16, pp. 265-283).

Documentation of the Python library Tensorflow. The library serves as a backend for machine learning development natively in python. Used throughout my methods and script.

Bayes, T. (1763). LII. An essay towards solving a problem in the doctrine of chances. By the late Rev. Mr. Bayes, FRS communicated by Mr. Price, in a letter to John Canton, AMFR S. Philosophical transactions of the Royal Society of London, (53), 370-418.

First literature on mathematically and statistically modeling thought and decision making. The essay set forth a new field of mathematics that serves as the background for all machine learning processes and methods. Used in literature review and script.

Bengio, Y., Simard, P., & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. IEEE transactions on neural networks, 5(2), 157-166.

Reviews methods in which to optimize recurrent neural networks so the learning gradient does not degenerate over time. Bengio's identification of the LSTM's efficiency in solving long-term lag problems drove the thought behind much of the methods. Used in methods section and script.

Chollet, F. (2018, March 27). Keras (Version 2.2.4) [Program documentation]. Retrieved 2018, from https://keras.io/

Documentation of the Python Machine Learning library Keras. The code allows for easy construction of powerful and complex machine learning models. Used in methods section and script.

Ehlers, J. F. (2001). Rocket science for traders: digital signal processing applications (Vol. 112). John Wiley & Sons.

Supplies a history of arbitrage trading techniques and the expansion of computational finance. Used in literature review.

Fama, E. (1970). Efficient Capital Markets: A Review of Theory and Empirical Work. The Journal of Finance, 25(2), 383-417.

A deep look at how efficient the market really is, and the potential sources of error in the assumptions made by EMH. Used in conclusions section.

Fischer, T., & Krauss, C. (2018). Deep learning with long short-term memory networks for financial market predictions. European Journal of Operational Research, 270(2), 654-669.

Revolutionary literature that compares the effectiveness of long short-term memory agents in performing statistical arbitrage on the S&P 500 index. Much of the methods this paper outlines are derivatives of theirs. Used throughout the paper and script.

Fox, J., & Sklar, A. (2009). The myth of the rational market: A history of risk, reward, and delusion on Wall Street (p. xi). New York: Harper Business.

Supplied a concise history of computational finance. Used in literature review section.

Goodkin, M. (2012). The Wrong Answer Faster: The Inside Story of Making the Machine that Trades Trillions. John Wiley & Sons.

A long-winded book outlining the history of trading systems. Used in literature review section.

Green, J., Hand, John R. M., Zhang, X. F., (2013). The supraview of return predictive signals. Review of Accounting Studies 18 (3), 692–730

A survey on how non-textual based market anomalies can be used as statistically significant markers of market direction. Used in conclusions section.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8),1735-1780.

First described the processes, algorithms for, and functionalities of the LSTM. Processes detailed are the basis of the methods. Used in literature review section and script.

Hochreiter, S., & Schmidhuber, J. (1997). LSTM can solve hard long-time lag problems. In Advances in neural information processing systems (pp. 473-479).

Published in conjunction with the previous title, described the processes, algorithms for, and functionalities of the LSTM, as well as the problems the algorithm is most effective at addressing. Processes detailed are the basis of the methods. Used in literature review section and script.

Huang, W., Nakamori, Y., & Wang, S. Y. (2005). Forecasting stock market movement direction with support vector machine. Computers & Operations Research, 32(10), 2513-2522.

An early effective attempt at using machine learning to predict financial markets based entirely on past performance. Background research.

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. Computing in science & engineering, 9(3), 90-95. doi: :10.1109/MCSE.2007.55

Documentation of the Python library MatplotLib, a library that makes data visualization possible. Used in methods section and script.

Jacobs, H., (2015). What explains the dynamics of 100 anomalies? Journal of Banking & Finance, 57, 65–85.

A survey on how non-textual based market anomalies can be used as statistically significant markers of market direction. While Jacobs' surveys identify issues, some of the methods are computationally expensive and ill-advised. Used in literature review section.

Karpathy, A. (2015). The unreasonable effectiveness of recurrent neural networks. Andrej Karpathy blog.

A helpful and explanatory blog document explaining the math behind LSTMs and other recurrent neural networks. Background research.

Krauss, C., Do, X. A., & Huck, N. (2017). Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P 500. European Journal of Operational Research, 259(2), 689-702.

Revolutionary literature comparing the effectiveness of several different machine learning algorithms in performing statistical arbitrage on the S&P 500 index. Much of the methods and conclusions this paper outlines are derivatives of theirs. Used throughout the paper and script.

Levine, D. M. (2013). A day in the quiet life of a NYSE floor trader. Retrieved January 13, 2018.

Supplies insight into how frequently trades are made by automated insights, rather than human selection. Used in conclusions section.

Malkiel, B. G. (1973). A Random Walk Down Wall Street [By] Burton G. Malkiel. Norton.

Book that popularized the EMH, cathing media attention and public support. First proposes the famous “blind monkeys” analogy. Background research.

Markowitz, H. (1952). Portfolio Selection. The Journal of Finance,7(1), 77-91. doi:10.2307/2975974

This paper outlines the first strategies for computational finance. Used in literature review.

Marr, B. (2016). A Short History of Machine Learning-Every Manager Should Read. Forbes. URL: https://www. forbes. com/sites/bernardmarr/2016/02/19/a-shorthistory-of-machine-learning-every-managershould-read.

Supplied a brief timeline of machine learning discoveries, trends, and advancements in the financial sector. Used in literature review.

McDowell, H. (2018). Thomson Reuters closes deal with Blackstone, rebrands as Refinitiv. [online] The TRADE. Available at: https://www.thetradenews.com/thomson-reuters-closes-deal-blackstone-rebrands-refinitiv/ [Accessed 1 Mar. 2019].

Supplied an overview of the transfer of Reuter's Eikon to Refinitiv. Referenced in a footnote.

McKinney, W. (2010, June). Data structures for statistical computing in python. In Proceedings of the 9th Python in Science Conference (Vol. 445, pp. 51-56).

Documentation of the Python library Pandas, a package for data manipulation. Used in methods section and script.

Oliphant, T. E. (2006). A guide to NumPy (Vol. 1, p. 85). USA: Trelgol Publishing.

Documentation of the Python library NumPy, a library for higher mathematics and data manipulation. Used in methods section and script.

Porshnev, A., Redkin, I., & Shevchenko, A. (2013, December). Machine learning in prediction of stock market indicators based on historical data and data from twitter sentiment analysis. In 2013 IEEE 13th International Conference on Data Mining Workshops (pp. 440-444). IEEE.

An excellent modern research paper or financial forecasting that fails to bridge the gap between disciplines and has issues with input data because of this. Used in literature review.

Rubin, R. and Collins, M. (2015). How an exclusive hedge fund turbocharged its retirement plan. Bloomberg Business.

One of the only available accounts of how well a specific private, ML-based hedge fund performs and has performed historically. Used in literature review section.

Schumaker, R., & Chen, H. (2006). Textual analysis of stock market prediction using financial news articles. AMCIS 2006 Proceedings, 185.

An early effective attempt at the use of machine learning in the prediction of financial markets based entirely on textual analysis. Used in literature review section.

Schumaker, R. P., & Chen, H. (2009). Textual analysis of stock market prediction using breaking financial news: The AZFin text system. ACM Transactions on Information Systems (TOIS), 27(2), 12.

A more succinct version of the previous paper that describes specific textual indicators of market direction in depth. Background research.

Sharpe, W. F. (1966). Mutual fund performance. The Journal of business, 39(1), 119-138.

First groundwork theory describing an unpredictable market, supplying the basis for many stock market prediction strategies. Conflicting with the ideas of technical analysis, this paper examines the validity and strength of its claims. Used throughout the paper.

Siami-Namini, S., & Namin, A. S. (2018). Forecasting economics and financial time series: Arima vs. lstm. arXiv preprint arXiv:1803.06386.

Found LSTMs to outpace conventional financial algorithms but found no significant difference on epochs. Used as background research to support the use of the LSTM in this application, also used in results section.

Thomson Reuters Eikon. (2018a). [Daily Total Returns of Every Historical Constituent of the S&P500, 1998-2017]. Retrieved 2018 from https://eikon.thomsonreuters.com/index.html

Source of historical data on all historical constituents. Courtesy of the Reuters Eikon Sales Team via the Interdisciplinary Research Collaboratory at the University of California Santa Barbra. Used in methods section and script.

Thomson Reuters Eikon. (2018b). [List of all Join and Exit Dates of Historical Constituent of the S&P500]. Retrieved 2018 from https://eikon.thomsonreuters.com/index.html

Source of data for all historical constituents' joiner and leaver dates. Courtesy of the Reuters Eikon Sales Team via the Interdisciplinary Research Collaboratory at the University of California Santa Barbra. Used in methods section and script.

Turing, A.M. (1950). Computing machinery and intelligence. Mind, 59(236), 433-460.

First paper outlining concepts and definitions of Machine learning. Used in literature review section and script.

Xiong, R., Nichols, E. P., & Shen, Y. (2015). Deep learning stock volatility with google domestic trends. arXiv preprint arXiv:1512.04916.

Examines the effectiveness of LSTMs in the prediction of the direction of the S&P 500 using both historical data of the index, and discrete data on domestic search volume and trends of macroeconomics-related terms. Some processes and data benchmarks used in my script. Background research.

Zheng, A., & Jin, J (n.d.). Using AI to Make Predictions on Stock Market.

Names many of the different variables in the machine learning research field that make investigations so expansive. Used in literature review section and script.