Predicting stock returns using LSTM

Benjamin von Essen and Haohang Wu

***Abstract*—Deep learning has been in the spotlight during the recent years thanks to its ability to model complex non-linear phenomena, which has proved useful in the field of asset pricing, which historically has depended heavily on linear models. While the standard multilayer perceptron has shows promise for use in asset pricing, they lack a sense of temporal dependencies, which is problematic since most data used is inherently on time series form. Thus, an LSTM model might be more suitable, since it can learn temporal dependencies. In this project we develop a LSTM model with a novel loss function which we train to predict cumulative returns in the future, given the last years values of 34 different features, ranging from firm level ratios and trading data to large scale factors, for the current 30 stocks included in the DJIA, as well as 19 stocks that used to be included. Following a couple of different simple buying strategies based on the predictions of our network, we managed, in all cases tested, to outperform the equally weighted index of stocks, for out-of-sample data in our test dataset, which included the Covid-19 crisis. Analyzing the weights of our trained network, we identify that the main features that are used by the network are the 5 Fama French factors, momentum and financial ratios.**

***Index Terms*—Empirical asset pricing, deep learning, neural networks, LSTM, time varying risk premia, factor models.**

I. Introduction

Conventional empirical asset pricing literature relies heavily on techniques of time series and panel data analysis. In this project, we instead utilize *Long and Short Term Memory (LSTM)*, a model from the field of deep learning, to predict asset returns, which is analogous to measuring risk premia [1]. The aim of the project is to train a neural netowrk that can predict a given firms cumulative returns a number of days into the future, given a time series of 34 different features, and evaluate its performance by using its predictions in a few different trading strategies. Two things that are noteworthy about our implementation is that we use a novel loss function, which penalizes predictions with the wrong sign more harshly, and that we instead of predicting daily returns try to predict cumulative returns for each forecasted day. Last but not least, we aim to measure the importance of the given features. The firms used for training, validation and testing consists of the current 30 firms included in the DJIA, as well as 19 firms that used to be included. All the code used in this project can be found in the following GitHub repository: `https://github.com/tekniktomten/LSTM-for-predicting-returns`.

II. Background

A. Empirical asset pricing theory

The most important goal of asset pricing is "to understand the behavior of risk premia" (Gu et al., 2020). Since the introduction of *capital asset pricing model (CAPM)* by Sharpe (1964), Lintner (1965) and Black (1972), asset pricing literature has embarked on the journey to this goal and generated plenty of theoretical asset pricing models based on CAPM.

$$E(R_i) - R_f = b_i[E(R_M) - R_f]$$

Merton (1973) develops the *Intertemporal CAPM (ICAPM)* to put CAPM under the dynamic environment and Breeden (1979) replaces aggregate wealth with aggregate consumption to measure the model's beta, which is now known as *Consumption CAPM (CCAPM)*. These models provide researchers abundant topics to explore. In principle, researchers regress stock returns on their betas to test CAPM. Since beta cannot be known beforehand, it needs to be estimated and the results are prone to errors. Fama & MacBeth (1973) devise a two-pass test to cope with this problem and use portfolios rather than individual stock to enhance the precision of the estimates of beta. However, the empirical records fail to provide evidence to support CAPM (Fama & French, 2004). Roll (1977) argues that the market portfolio in the CAPM should include every single possible available asset, but it is not possible to observe returns of such a portfolio .Malkiel & Fama (1970) points out the *joint hypothesis problem* which states that when testing CAPM, it is impossible to distinguish whether the market is not efficient, or the model is wrong because the regression simultaneously tests market efficiency and the CAPM.

Apart from these tests there are many empirical contradictions of the CAPM, which doubt market premium to be the only risk factor to explain the stock returns. Basu (1977) raises P/E ratio as an indicator of future stock performance and Banz (1981) finds that smaller firms have higher risk premium than larger firms on average. These findings have encouraged researchers to explore other asset pricing models to explain expected returns of stocks. Fama & French (1992, 1996) pioneer this exploration and introduce value (*HML*) and size (*SMB*) factor into their famous *three-factor model* (hereinafter *FF3*).

$$E(R_i) - R_f = b_i[E(R_M) - R_f] + s_i E(SMB) + h_i E(HML)$$

They have demonstrated that their model could capture much of the cross-sectional variation in average stock returns and most anomalies can also be attributed to the value and size factors. The *FF3* model "brought order once again with size and value factors" (Cochrane, 2011).

Since then, hundreds of papers have tried to test the *FF3* model using different data sets and they have found many anomalies during this process. These anomalies "include among many others, momentum, accruals, equity issues and other accounting-related sorts, beta arbitrage, credit risk, bond and equity market-timing strategies, foreign exchange carry trade, put option writing, and various forms of 'liquidity

[1] In this report, we will follow Gu et al. (2020) to use "expected return", "risk premia" and "excess return" interchangeably. All of them refer to the conditional expected returns.

provision'" (Cochrane, 2011). Numerous factors have been devised to correspond to these anomalies. Harvey et al. (2016) study 316 factors in the literature, including firm features and common factors, for explaining the cross-sectional expected returns. Researchers today are mainly trying to find the real significant factors from this factor zoo and the economic logic behind the regression results. However, as the number of factors approaches the observations and lots of factors are highly correlated, it is difficult for traditional econometric toolbox to render meaningful results and new methods need to be introduced instead.

B. Machine learning and Neural Networks

Generally speaking, machine learning is the study of algorithms which improve automatically through experience and the use of data (Mitchell et al., 1997). It helps us to tackle problems which are too complex to solve with conventional methods written and designed by human beings. As one of the most popular methods in machine learning, neural networks borrows a lot of concept from neurons in our brains. By simulating the information processing among neurons, neural network methods gained popularity due to the fact that they are able to model data using limited assumptions on the underlying distribution and capture patterns which are too difficult for many other algorithms to find. One important feature neural networks enjoy is that we do no need to make assumption of the distribution of data since the neural networks itself discovers it.

One of the most basic neural network models is called *deep feedforward networks*, also *feedforward neural networks*, or *multilayer perceptrons (MLPs)*. In this model, information tends to flow through the input function and intermediate computations and finally to the output. There are no feedback connections in which outputs of the model are fed back into itself. It consists of one input layer followed by a number of hidden layers and ends with one output layer, as illustrated in Figure 1. When feedforward neural networks are extended to include feedback connections, they are called *recurrent neural networks*, which are powerful in dealing with sequential data in the financial industry.

1) Feedforward neural network: The general formula for a feedforward neural network (MLP) is constructed as follows:

initial input:

$$x_0 = (1, z_1, ..., z_N)'$$

hidden layer:

$$h_l = \sigma\left(\mathbf{W}_l^{(hh)} h_{l-1} + b_h\right) \tag{1}$$

with final output:

$$g(\mathbf{Z}; \mathbf{W}) = \mathbf{W}_{L-1}^{(yh)} h_{L-1} + b_y \tag{2}$$

where

- l is the number of hidden layers $l = 1, ..., L-1$
- h_l is the output of hidden layer l
- σ is the activation function which adds nonlinearity into the model

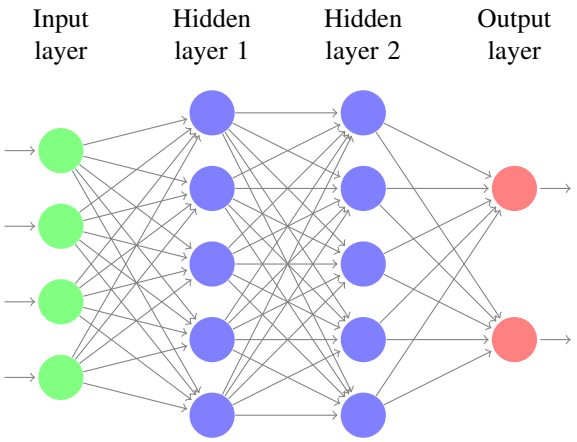

Figure 1. A multilayer perceptron for binary classification with 2 hidden layers.

- $\mathbf{W}^{(hh)}$ and $\mathbf{W}^{(yh)}$ are the corresponding weight matrices
- b_h and b_y are the biases

2) Recurrent neural networks (RNNs): The RNN family of network architectures is specialized in predicting sequential data. Much like humans' decision process with sequential information, RNNs introduces the notion of time to the feedforward neural networks. That said, RNNs connects the hidden layers in the feedforward neural networks to each other and exhibit a temporal behavior. Figure 2 depicts the structure of a simple RNN and its unfolded representation. Instead of interpreting the network as cyclic, one can view it as a feedforward neural network with one layer per time step and share weights across time steps (Cong et al., 2021).

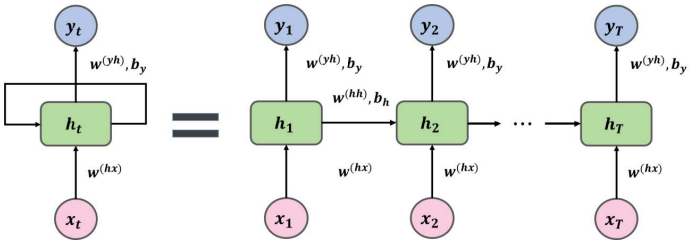

Figure 2. Simple RNN structure

Unlike feedforward neural networks, the hidden layer of RNN is computed as:

$$h_t = \sigma\left(\mathbf{W}^{(hx)} x_t + \mathbf{W}^{(hh)} h_{t-1} + b_h\right) \tag{3}$$

with final output:

$$g(\mathbf{Z}; \mathbf{W}) = \mathbf{W}^{(yh)} h_t + b_y \tag{4}$$

where

- x_t is the input at time t
- h_t is the output of hidden layer at time t
- σ is the activation function which adds nonlinearity into the model
- $\mathbf{W}^{(hx)}$, $\mathbf{W}^{(hh)}$ and $\mathbf{W}^{(yh)}$ are the corresponding weight matrices
- b_h and b_y are the biases

When using a back propagation algorithm to calculate the gradients of neural networks, its ability to tune parameters is often limited by *gradient vanishing/exploding* problems. For RNNs, these problems are exacerbated by the recurrent connections. By the chain rule, the gradients decrease/increase as we propagate down to the initial layer, which in case of vanishing/exploding completely stops the neural networks from updating the weight matrices. To solve these problems, long short term memory (LSTM) neural networks can be used instead.

3) Long and Short Term Memory: Long and Short Term Memory (LSTM) is one variant of basic *recurrent neural networks (RNNs)* capable of learning long-term dependencies, which is something the basic RNN struggles to do. They were initially developed to solve the problem of a vanishing/exploding gradient.

The LSTM adds a *memory cell* from input layer to hidden layer in RNN that have an internal recurrence (a self-loop), in addition to the outer recurrence of the RNN (Goodfellow et al., 2016). As is shown in Figure 3, the data flow in and out of this *memory cell* are controlled by several *gates*, namely, *input gate, output gate and forget gate*. The forget gate is introduced to allow the network to choose which information to remember.

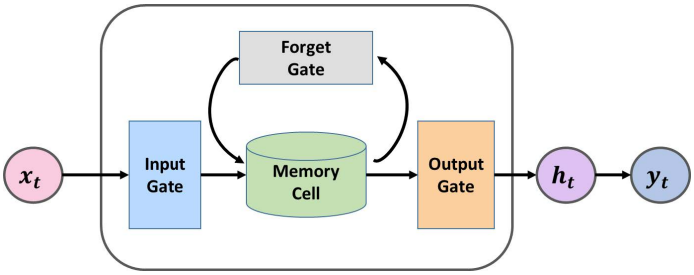

Figure 3. Illustration of LSTM structure

The basic formula for LSTM is calculated as follow:

$$f_t = \sigma\left(\mathbf{W}^{(fh)}h_{t-1} + \mathbf{W}^{(fx)}x_t + b_f\right) \quad (5)$$

$$i_t = \sigma\left(\mathbf{W}^{(ih)}h_{t-1} + \mathbf{W}^{(ix)}x_t + b_i\right) \quad (6)$$

$$o_t = \sigma\left(\mathbf{W}^{(oh)}h_{t-1} + \mathbf{W}^{(ox)}x_t + b_o\right) \quad (7)$$

$$c_t = f_t c_{t-1} + i_t tanh\left(\mathbf{W}^{(ch)}h_{t-1} + \mathbf{W}^{(cx)}x_t + b_c\right) \quad (8)$$

$$h_t = o_t tanh(c_t) \quad (9)$$

with final output:

$$g(\mathbf{Z}; \mathbf{W}) = \mathbf{W}^{(yh)}h_t + b_y \quad (10)$$

where

- x_t is the input at time t
- f_t is the forget gate at time t
- i_t is the input gate at time t
- o_t is the output gate at time t
- c_t is the memory value at time t
- h_t is the output of hidden layer at time t
- σ is the sigmoid activation function which takes the form as:

$$\sigma(t) = \frac{e^t}{1 + e^t}$$

- $tanh$ is the tanh activation function which takes the form as:

$$tanh(t) = \frac{e^t - e^{-t}}{e^t + e^{-t}}$$

- $\mathbf{W}$ are the corresponding weight matrices
- b are the biases

C. Asset pricing via machine learning

Empirical asset pricing aims to measure the conditional expectation of future excess return and it is theoretically a prediction problem. Machine learning is an ideal tool to solve such problems. Besides, the sheer number of factors and the ambiguity of the functional forms of asset pricing model further increase the difficulties for traditional models to generate satisfying prediction. Machine learning methods have advantages on these areas. Modeling the risk premium using machine learning methods is a natural step forward.

During these years, there is an emerging literature that uses machine learning tools in empirical asset pricing. In their comprehensive work Gu et al. (2020) conduct a comparative analysis of machine learning methods to forecast the expected return of individual US stocks. They have found that nonlinear models substantially improve predictions of expected returns compared with conventional linear models. Amongst the nonlinear models they use, deep neural network with three hidden layers generates the best forecast.

The neural network models which Gu et al. (2020) use rely on conventional feed-forward neural network by assuming the conditional expected return only uses short history information (one period ahead). To capture momentum and reversal effect, their model instead takes some trend factors into their prediction. However, these predetermined factors cannot fully reflect the serial dependence known in asset returns. Chen et al. (2019) and Cong et al. (2021) take the natural step forward to use *recurrent neural networks (RNNs)* to model the serial dependence. They have demonstrated that RNNs have shown better out-of-sample performance.

III. Methodology

A. Data

For our dataset, we decided to include all 30 stocks that are currently in the Dow Jones Industrial average, plus all stocks that have been dropped from the index since 1991 (if they still exist and are not included in another way because of mergers or carve-outs etc.), which resulted in 49 stocks in total. The reason for also including the dropped stocks was to introduce some more variability in the returns and make sure that there also exists some worse performers. This made our full portfolio consists or almost exclusively large cap stocks, except for some fallen angels such as Kodak. The full list of stocks can be found in Table II in the Appendix.

The data used is collected from the sources, as described below.

- Yahoo Finance provided daily Adjusted Closing prices and Volume.
- Kenneth French's Website provided us with the daily Fama French 5 factors model's factors, the risk free rate and the momentum factor, all for North America.
- Compustat was provided by Services-WRDS (2021), which WRDS describes as "a collection of most commonly used financial ratios by academic researchers. There are in total over 70 financial ratios grouped into the following seven categories: Capitalization, Efficiency, Financial Soundness/Solvency, Liquidity, Profitability, Valuation and Others." They also state that "In addition, in order to make sure that all data is publicly available at the monthly time stamp, we lag all observations by two months to avoid any look ahead bias".

The data was then pre-processed, which included

- Turning adjusted closing prices into returns to make them more stationary.
- Normalizing the volumes by the first recorded volume and then into the 7 day rolling average.
- Discarding non quantitative and not relevant features from the Compustat data.
- Discarding features identifying the firm or the date from the Compustat data (although these were used for joining the different datasets).
- Discarding all features with more than 1% NaN values.
- Replacing the remaining NaNs with zeros.
- Turning the returns used as labels, Y, into cumulative returns since the last input day.

In order to align with the deep learning approach of trusting the network to create its own useful representations of the data, we opted to not construct any new features and left the data as raw as possible. Normalizing the data is however very common in deep learning, but this could be omitted for most variables as they were already almost the same size (because of them mostly being ratios or on index form).

The datasets were then joined by firm and by date. Since the Compustat data had lower frequency than the rest, the missing dates were forward filled, turning them into daily (although not always changing) data.

The datasets were then split into training, validation and test samples. Our whole dataset contained three major financial crises, the dotcom bubble, the financial crisis and Covid-19. We decided to distribute these so that training covers the first two crises, validation gets none and test gets Covid-19. While this makes our validation set worse, we decided that it is more important that we get enough training data and that the test subset contains a crisis, since how well a model handles that is essential for long term profitability (if the model is used for making investment decisions). Our full dataset spans all the years from 1998 to 2020 (including 2020), and our test subset starts 2018-02-21.

After all the pre-processing, our dataset has the dimensions seen in Table I, each being a 3D features with the following dimensions: (number of samples, time steps per sample, number of features). The labels, Y, had only one feature, cumulative adjusted close returns from the latest input date, but 63 time steps. This means that for each input sample, the neural network tries to predict the cumulative returns for each of the coming 63 trading days.

Table I
DATA DIMENSIONALITY

Dataset	Dimensions
trainX	(196000, 253, 34)
trainY	(196000, 63, 1)
valX	(33663, 253, 34)
valY	(33663, 63, 1)
testX	(29106, 253, 34)
testY	(29106, 63, 1)

B. Neural network

The neural network used in this project was implemented in Keras, which is a frontend for using Tensorflow in Python. An LSTM was chosen as our network architecture of choice, as we believe its ability to capture temporal relationships is crucial for properly modelling time varying risk premia. Our neural network was configured to have the following characteristics:

- The input layers is of size (253, 34), as each input consists 34 features per day the last year (253 trading days).
- The input layers is followed by one LSTM layers with 100 units.
- The LSTM layer is then followed by an output layer, which consists of a Dense layer (i.e. a standard layer, like the ones used in MLPs), which learns to map the output from the LSTM layer to the desired output, i.e. the cumulative return from the last input day for the coming 63 days (3 months), see section III-B1.
- Lastly, the output layer is passed through a reshape layer which changes it from size a vector of size (63) to a equivalent matrix of size (63, 1).
- Adam is used for training.
- A custom loss function is used, see section III-B2.
- All other parameters are left to Keras' default values.

Worth noting is that we in this project focus on predicting returns and not excess returns (i.e. returns minus risk free rate). Although most theory is concerned with excess returns, we do not think this is problematic, since returns are a good proxy for excess returns. If you can predict one then you can predict the other quite well, since the risk free rate in general moves slower than returns and is relatively small. If one wanted to train the same model to instead predict excess returns, exactly the same model could be used, but risk free rate would need to be subtracted from the returns when training.

1) Predicting cumulative returns: A common approach to predicting asset returns is to, given input data up until one date, estimate the return a number of days ahead, e.g. always trying to predict the total returns for the coming month. While this approach makes sense for a MLP, it seems lackluster when used in conjunction with an LSTM. As we have daily input data, we want the output data to also be daily, so that the LSTM can estimate the continued time series of returns with equally distanced (in time) steps. Thus, a seemingly reasonable approach is to predict returns each day. To then decide upon

the expected return over a forecast horizon, the cumulative returns would then need to be calculated. This approach was the one we first tried, but we soon realized that it might be better to train the LSTM to output this cumulative returns directly. This has two advantages, first, it aligns our goals with the neural networks loss function and second, it makes each single days returns less important, possibly making it easier to catch trends. Thus, we decided upon trying to predict *cumulative* returns from the last input day, meaning that in the common case of positive returns, the returns should increase with the forecast horizon. However, one could argue that this turns the desired output non-stationary and growing with time, making it harder for the network to learn. Having said that, we object to this line of reasoning; it is clear that the first output will stay the same and thus will not be affected in terms of stationary, and if we recall that LSTMs share hidden states over time, it is not at all unreasonable that it picks up the cumulative effect and in the case of strictly positives returns learns the growing returns, in each step increasing the returns indicated by the last steps hidden state.

2) Custom loss function: The standard loss function for regression type prediction problems is MSE, or sometimes MAE. Thus, in our initial experiments we used MSE as our loss function. But, after getting agitated by our network recommending us to buy stocks which went down in value, we decided to try to prevent this behaviour. While it is often a bad idea to tamper with the loss function, as it is important that it satisfies a number of criteria, we decided that it was worth to explore changing the loss function to penalize predictions of the wrong sign more. After a couple of failed attempts, such as an asymmetric loss which made the network always guess slightly less than zero, we found a modified loss function which gave satisfactory results. This novel loss function is described in equation (11).

$$loss = \begin{cases} if\ correct\ sign & MSE \\ else & MSE \cdot 1.5 \end{cases} \tag{11}$$

C. Performance evaluation

We devised a number of evaluation criteria, and one measure of variable importance.

- Correlation between true and predicted returns for different forecast horizons, for the training dataset.
- Correlation between true and predicted returns for different forecast horizons, for the validation dataset.
- Correlation between true and predicted returns for different forecast horizons, for the test dataset.
- Portfolio development over time when, each month, buying the top 10 predicted stocks, with a forecast horizon of one month, and selling them next month versus buying all stocks equally weighted. Calculated on the test dataset.
- Portfolio development over time when, each day, buying the top 10 predicted stocks, with a forecast horizon of one day, and selling them the next day versus buying all stocks equally weighted. Calculated on the test dataset.
- Portfolio development over time when, each month, buying the all the stocks with positive predicted returns, for a forecast horizon of one month, and selling them next month versus buying all stocks equally weighted. Calculated on the test dataset.
- Sum of the absolute values of the weights per feature, which gives us a crude measurement of feature importance.
- Calculating the maximum weight per unit, finding the corresponding feature and that weights percentage of the whole (in absolute value terms) and ranking them in order of percentage of the whole, thus achieving a ranking of the units which the most powerful feature connections and their corresponding strongest feature.

IV. Results

In Figure 4 we see how the loss developed per over epochs when training. Figure 5 to 7 show our correlation measures (note the different scales) and Figure 8 to 10 we can see the promising results of the performance tests. The results of our feature importance measures can be seen in Figure 11 and 12.

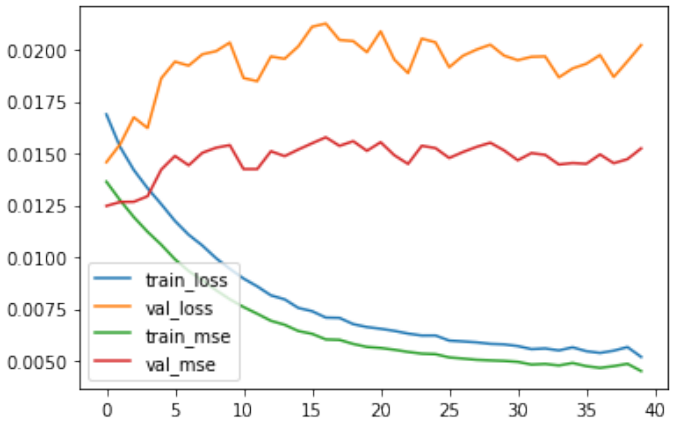

Figure 4. Loss and MSE for each epoch of training

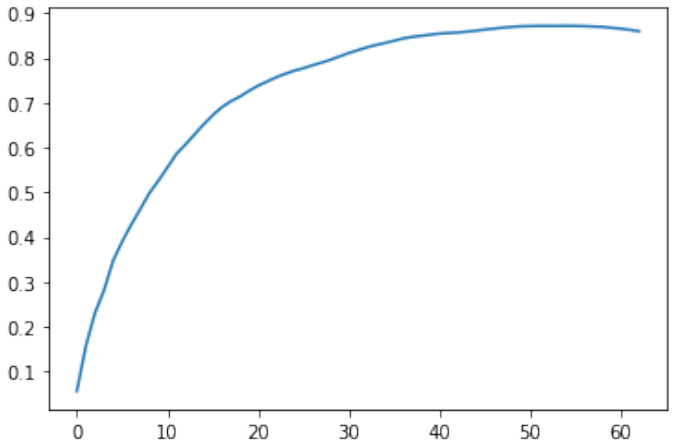

Figure 5. Correlation between true and predicted returns for different forecast horizons, for the training dataset.

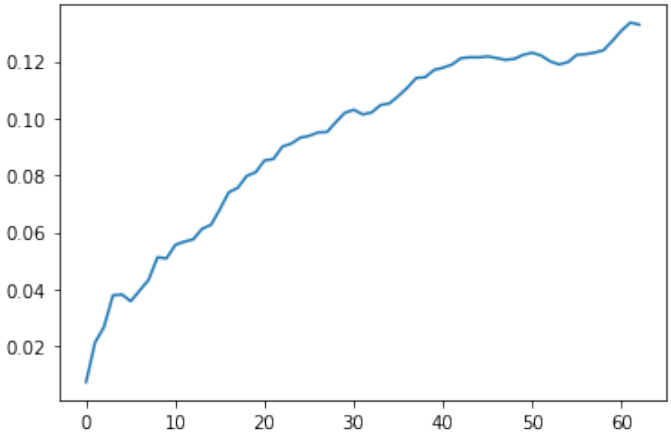

Figure 6. Correlation between true and predicted returns for different forecast horizons, for the validation dataset.

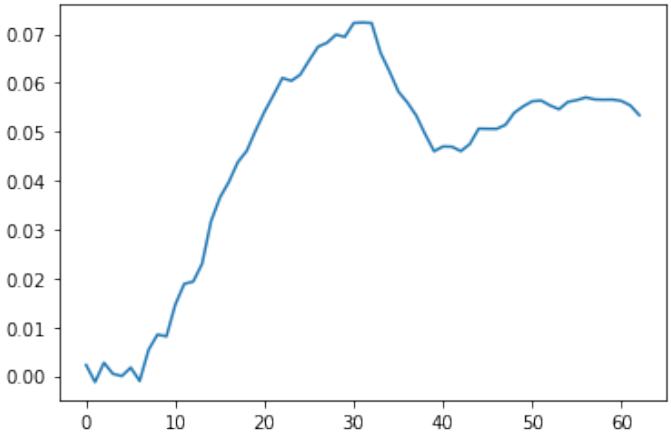

Figure 7. Correlation between true and predicted returns for different forecast horizons, for the test dataset.

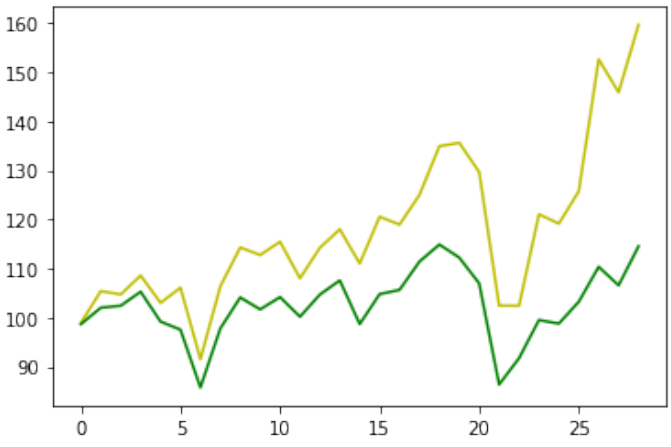

Figure 8. Buying top 10, monthly. Portfolio development over time when, each month, buying the top 10 predicted stocks, with a forecast horizon of one month, and selling them next month (in yellow) versus buying all stocks equally weighted (in green).

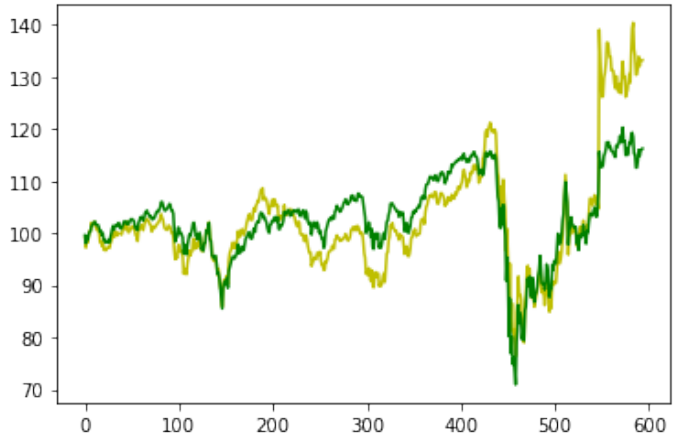

Figure 9. Buying top 10, daily. Portfolio development over time when, each day, buying the top 10 predicted stocks, with a forecast horizon of one day, and selling them the next day (in yellow) versus buying all stocks equally weighted (in green).

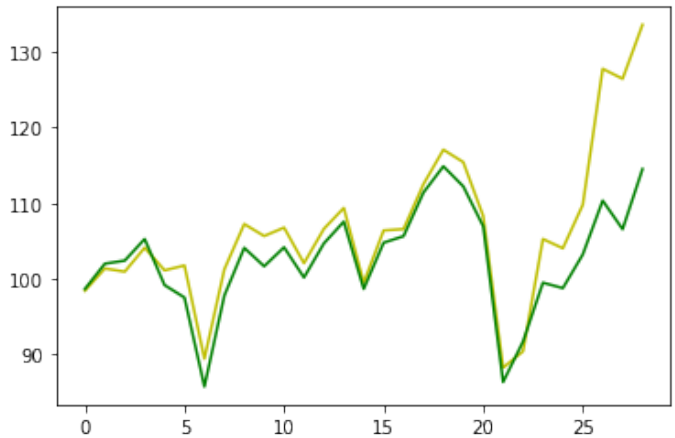

Figure 10. Buying all positive predictions, monthly. Portfolio development over time when, each month, buying the all the stocks with positive predicted returns, for a forecast horizon of one month, and selling them next month (in yellow) versus buying all stocks equally weighted (in green).

V. Discussion

A. Training and evaluation

As seen in Figure 4, training loss decreased fast in the beginning but then mostly plateaued after around 30 epochs, which indicated that we trained long enough. One thing that instantly grabs ones attention is the non-decreasing validation loss (which even seems to increase during the first 5 epochs)! Does our model overfit? The answer is maybe, but we do not believe that it overfits in a harmful way. It is natural that it learns in sample data better than out of sample data. Furthermore, one really important thing to consider is that, due to data limitations, our validation set was fundamentally flawed as it did not contain a crisis. Thus, it is quite possible that the model learned to handles these crises, which resulted in lower training loss but no change in the validation loss. Furthermore, since the validation loss also plateaued and did not increase (except for in the very beginning) we make the conclusion that training for 40 epochs did not lead to

more severe overfitting than training for 5 epochs and that the network hopefully learned to recognise important patterns (possibly connected to times of financial distress) during later epochs.

We find it very interesting that the correlation for different forecast horizons takes the form of the function $1 - e^{-x}$ for the training dataset. We believe this to be connected to the cumulative nature of the output.

We also see that the prediction seems to grow stronger with a longer forecast horizon for the validation dataset, but that the test dataset has the strongest correlation at 30 days. We believe this to be connected to the fact that there is no crisis in the validation dataset, making it more similar over time, and that the Covid-19 crisis in the test dataset makes it hard to predict returns too far into the future, as one must react to the crisis quite quickly.

Although the correlation plots are useful to measure performance for different forecast horizons, we want to advice the reader to be a bit cautious of this measurment, as it is not the best measure of performance. Furthermore, the cumulative nature of the prediction make it inherently easier to do forecast farther into the future.

B. Portfolio performance

We were happily surprised that our network outperformed the equally weighted index when predicting 21 days ahead, both when buying the top 10 predictions (which performed best) as well as when buying everything with positive predicted returns. Even though the daily prediction did outperform the index eventually, it overall performed less consistently. While the 21 days ahead strategies almost always lied over the index, the 1 day ahead strategy often lied below. This is in line with our expectations as well as the correlation plots.

Overall, our conclusion is that strategies based on the LSTM networks predictions performs better than buying an equally weighted index of stocks.

C. Features that matter

For the 34 features used in our model, we categorize these features in line with *WRDS Industry Financial Ratio (WIFR)*. Table III in Appendix demonstrates the complete list of features and corresponding categorization. As is shown in Figure 11 and 12, we have evaluated the relative importance of these features by the absolute value of their weights and individual performance in units in the LSTM model.

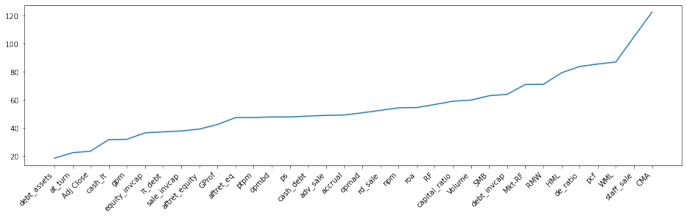

Figure 11. Variable importance in LSTM

Under *Efficient Market Hypothesis (EMH)*, all information should be reflected in the stock price, thus stock price follows

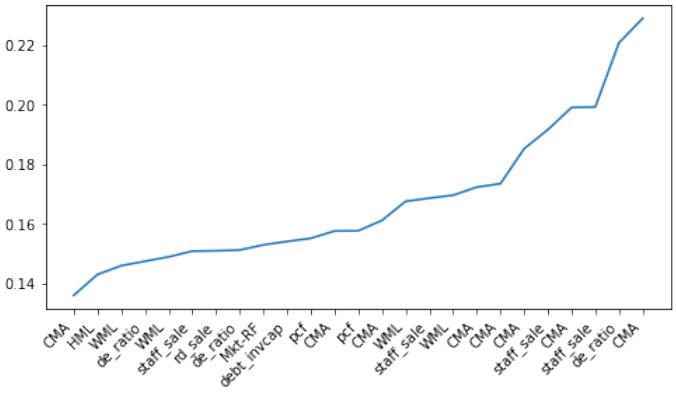

Figure 12. The 20 LSTM units with the strongest single feature (in percentage of the whole on the Y-axis) and the corresponding feature (on the X-axis)

random walk process and stock return is unpredictable. In our model, adjusted stock returns (Adj Close) performs rather poor to forecast stock returns, while volume shows rather strong predictive power, indicating technical analysis might still provide some guidance in stock investment. This would violate the EMH theory.

In general, the most influential category of features is Fama-French 5 factors in our model, all ranking in the top 10 variables. Our findings have supported what Fama & French (2015) have argued, that size (SMB), value (HML), profitability (RMW), and investment patterns (CMA) would have a better capability to capture stock returns. Besides, for the Dow Jones Industrial Average Index, its components are mostly large-cap stocks. Our stock selection would effectively protect our model from dealing with small-cap stocks and avoid the predication failure which Fama and French have encountered in their original paper. The high frequency of investment pattern factor (CMA) shown in Figure 12 also highlights the necessity of introducing this factor into the prediction model.

Apart from Fama-French 5 factors, momentum factor (WML) also plays an important role in predicting stock returns in our model. Even controlling for the temporal dependence in individual stocks, momentum factor is still quite persistent in our LSTM model.

Labor expenses over sales ratio (staff_sale) turns out to be the second most influential feature in our forecast. High labor expenses might indicate that the firm are willing to invest in their employees and this would attract more talented people to join in, which would provide the firm advantages over its competitors and improve its performance.

Research and development expenses over sales (rd_sale) is also an influential feature in our model and this is in line with the findings of Chan et al. (2001) and Eberhart et al. (2004), R&D expenses are beneficial investments, and those stocks with high R&D expenses would gain positive excess returns following these expenses.

Debt-to-equity ratio (D/E), which is a classical measure of solvency, was also found to be a very influential predictor, as was Price-to-Cash Flow Ratio (P/CF), which is a common multiple.

It should be noted that financial ratios on profitability have experienced rather bad performance in our model, with most factors listing below the average. HML measures the return gap between value stocks and growth stocks and it might serve as a proxy for the fundamental difference between these two kinds of stocks. One important difference is profitability, with value stocks typically more profitable than growth stocks. In our models, HML performs rather good and it might capture the same information which profitability ratios account for. This would partly explain profitability ratios' bad performance.

Modern view argues that predictability of stock returns does not necessarily indicate that market is inefficient and stock price moves on the discount factors. Our model has demonstrated that one can use certain factors to forecast stock returns and generate time-varying risk premia.

D. The flaws of our approach and possible future research

While we are happy with the outcome of our project, there are a lot of room for improvement. First of all, our model focuses on the temporal dependence of individual stocks and does not consider the cross-asset dependence among these stocks. In reality, the returns of different stocks are often correlated and have loading on common factors. To capture these correlation, we could introduce cross-asset attention network, which touches some cutting-edge neural networks (Cong et al., 2020). Furthermore, because of the limited scope of this project and a limited computational capacity, there are several flaws in our methodology that could be improved. As ten epochs took around one hour to train, we decided to have static training, validation and tests sets. While this is a common approach for most machine learning tasks, time series prediction is often better evaluated by having a sliding windows of training, validation and tests sets. While that approach lends it self to more lifelike evaluation of prediction performance, it would have significantly increased the time it took to train and evaluate the network. We do however recommend that approach for further research and recommend that either a stronger GPU (than the Nvidia K80 we had) is used or that the project is carried out over a longer time period. One closely related problem is that of the validation set not including a financial crisis, which makes it worse for evaluating the level of overfitting. This could have been avoided is data spanning a larger time period was used, but this would, like introducing sliding windows, increased the computational burden of the project.

E. Does our model generalize?

While we have reasons to believe that our model generalizes well for our selection of stocks, such as the good portfolio returns for the test dataset as well as results aligning with earlier papers, further evaluation would be necessary if one wanted to deploy this model into the real world. Furthermore, we had a quite limited selection of stocks. If this helped our model or made it harder is hard to say, but we believe that using this model on stocks not included in the training dataset should be done with caution.

F. Does our modifications help?

It is hard to say if our novel loss function and predicting cumulative returns instead of daily returns lead to any improvement. To reach a conclusion about that, multiple networks would have to be trained and compared. However, since we got good results, we draw the conclusion that they work well, for this specific setup. Furthermore, we note that the novel loss function and the standard measure MSE are heavily correlated, but with some small deviations and a gap between them, as seen in Figure 4.

VI. Conclusion

In this project, we overview the empirical asset pricing history and the application of machine learning methods in predicting stock returns. We develop an LSTM model with a novel loss function and train it to predict cumulative stock returns. Following a couple of different simple buying strategies based on the predictions of our network, we manage, in all cases tested, to outperform the equally weighted index of stocks, for out-of-sample data in our test dataset, which included the Covid-19 crisis. Thus, our conclusion is that strategies based on the LSTM networks predictions performs better than buying an equally weighted index of stocks. Analyzing the weights of our trained network, we identify that the main features that are used by the network are the 5 Fama French factors, momentum and financial ratios.

Acknowledgment

The authors would like to thank Professor Adrien d'Avernas for his insightful lectures on asset pricing theories. The authors would also like to thank Wharton Research Data Services (WRDS), Yahoo Finance and Professor Kenneth R. French for providing the datasets which made this project possible.

References

Banz, R. W. (1981). The relationship between return and market value of common stocks. *Journal of financial economics*, *9*(1), 3–18.

Basu, S. (1977). Investment performance of common stocks in relation to their price-earnings ratios: A test of the efficient market hypothesis. *The journal of Finance*, *32*(3), 663–682.

Black, F. (1972). Capital market equilibrium with restricted borrowing. *The Journal of business*, *45*(3), 444–455.

Breeden, D. T. (1979). An intertemporal asset pricing model with stochastic consumption and investment opportunities. *Journal of Financial Economics*, *7*(3), 265–296.

Chan, L. K., Lakonishok, J., & Sougiannis, T. (2001). The stock market valuation of research and development expenditures. *The Journal of finance*, *56*(6), 2431–2456.

Chen, L., Pelger, M., & Zhu, J. (2019). Deep learning in asset pricing. *Available at SSRN 3350138*.

Cochrane, J. H. (2011). Presidential address: Discount rates. *The Journal of finance*, *66*(4), 1047–1108.

Cong, L. W., Tang, K., Wang, J., & Zhang, Y. (2020). Alphaportfolio for investment and economically interpretable ai. *Available at SSRN 3554486*.

Cong, L. W., Tang, K., Wang, J., & Zhang, Y. (2021). Deep sequence modeling: Development and applications in asset pricing. *The Journal of Financial Data Science*, *3*(1), 28–42.

Eberhart, A. C., Maxwell, W. F., & Siddique, A. R. (2004). An examination of long-term abnormal stock returns and operating performance following r&d increases. *The Journal of Finance*, *59*(2), 623–650.

Fama, E. F., & French, K. R. (1992). The cross-section of expected stock returns. *the Journal of Finance*, *47*(2), 427–465.

Fama, E. F., & French, K. R. (1996). Multifactor explanations of asset pricing anomalies. *The journal of finance*, *51*(1), 55–84.

Fama, E. F., & French, K. R. (2004). The capital asset pricing model: Theory and evidence. *Journal of economic perspectives*, *18*(3), 25–46.

Fama, E. F., & French, K. R. (2015). A five-factor asset pricing model. *Journal of financial economics*, *116*(1), 1–22.

Fama, E. F., & MacBeth, J. D. (1973). Risk, return, and equilibrium: Empirical tests. *Journal of political economy*, *81*(3), 607–636.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press. (`http://www.deeplearningbook.org`)

Gu, S., Kelly, B., & Xiu, D. (2020). Empirical asset pricing via machine learning. *The Review of Financial Studies*, *33*(5), 2223–2273.

Harvey, C. R., Liu, Y., & Zhu, H. (2016). ... and the cross-section of expected returns. *The Review of Financial Studies*, *29*(1), 5–68.

Lintner, J. (1965). Security prices, risk, and maximal gains from diversification. *The journal of finance*, *20*(4), 587–615.

Malkiel, B. G., & Fama, E. F. (1970). Efficient capital markets: A review of theory and empirical work. *The journal of Finance*, *25*(2), 383–417.

Merton, R. C. (1973). An intertemporal capital asset pricing model. *Econometrica: Journal of the Econometric Society*, 867–887.

Mitchell, T. M., et al. (1997). Machine learning.

Roll, R. (1977). A critique of the asset pricing theory's tests part i: On past and potential testability of the theory. *Journal of financial economics*, *4*(2), 129–176.

Services-WRDS, W. R. D. (2021). Wrds. (`wrds.wharton.upenn.edu`)

Sharpe, W. F. (1964). Capital asset prices: A theory of market equilibrium under conditions of risk. *The journal of finance*, *19*(3), 425–442.

APPENDIX

Table II
STOCKS USED IN THIS PROJECT

Ticker
MMM
AXP
AMGN
AAPL
BA
CAT
CVX
CSCO
KO
DOW
GS
HD
HON
IBM
INTC
JNJ
JPM
MCD
MRK
MSFT
NKE
PG
CRM
TRV
UNH
VZ
V
WBA
WMT
DIS
NAV
X
Z
GT
T
KODK
IP
MO
AIG
C
MDLZ
AA
BAC
HPQ
HPE
GE
RTX
XOM
PFE

Table III
LIST OF FEATURES AND CATEGORIZATION

#	Feature	Variable name	Category	Formula
1	Accruals/Average Assets	accrual	Other	Accruals as a fraction of average Total Assets based on most recent two periods
2	Adjusted Close Returns	Adj Close	Valuation	Close price today over last trading day
3	Avertising Expenses/Sales	adv_sale	Other	Advertising Expenses as a fraction of Sales
4	After-tax Return on Average Common Equity	aftret_eq	Profitability	Net Income as a fraction of average of Common Equity based on most recent two periods
5	After-tax Return on Total Stockholders' Equity	aftret_equity	Profitability	Net Income as a fraction of average of Total Shareholders' Equity based on most recent two periods
6	Asset Turnover	at_turn	Efficiency	Sales as a fraction of the average Total Assets based on the most recent two periods
7	Capitalization Ratio	capital_ratio	Capitalization	Total Long-term Debt as a fraction of the sum of Total Long-term Debt, Common/Ordinary Equity and Preferred Stock
8	Cash Flow/Total Debt	cash_debt	Solvency	Operating Cash Flow as a fraction of Total Debt
9	Cash Balance/Total Liabilities	cash_lt	Solvency	Cash Balance as a fraction of Total Liabilities
10	Conservative Minus Aggressive	CMA	Other	Average return on the two conservative investment portfolios minus the average return on the two aggressive investment portfolios
11	Total Debt/Equity	de_ratio	Solvency	Total Liabilities to Shareholders' Equity (common and preferred)
12	Total Debt/Total Assets	debt_assets	Solvency	Total Debt as a fraction of Total Assets
13	Long-term Debt/Invested Capital	debt_invcap	Capitalization	Long-term Debt as a fraction of Invested Capital
14	Common Equity/Invested Capital	equity_invcap	Capitalization	Common Equity as a fraction of Invested Capital
15	Gross Profit Margin	gpm	Profitability	Gross Profit as a fraction of Sales
16	Gross Profit/Total Assets	GProf	Profitability	Gross Profitability as a fraction of Total Assets
17	High Minus Low	HML	Valuation	Average return on the two value portfolios minus the average return on the two growth portfolios
18	Long-term Debt/Total Liabilities	lt_debt	Solvency	Long-term Debt as a fraction of Total Liabilities
19	Market premium	Mkt-RF	Other	Market is the return on North America's value-weight market portfolio minus the U.S. one month T-bill rate
20	Net Profit Margin	npm	Profitability	Net Income as a fraction of Total Assets
21	Operating Profit Margin After Depreciation	opmad	Profitability	Operating Income After Depreciation as a fraction of Sales
22	Operating Profit Margin Before Depreciation	opmbd	Profitability	Operating Income Before Depreciation as a fraction of Sales
23	Price/Cash flow	pcf	Valuation	Multiple of Market Value of Equity to Net Cash Flow from Operating Activities
24	Price/Sales	ps	Valuation	Multiple of Market Value of Equity to Sales
25	Pre-tax Profit Margin	ptpm	Profitability	Pretax Income as a fraction of Sales
26	Research and Development/Sales	rd_sale	Other	R&D expenses as a fraction of Sales
27	Risk-free rate	RF	Other	U.S. one month T-bill rate
28	Robust Minus Weak	RMW	Profitability	Average return on the two robust operating profitability portfolios minus the average return on the two weak operating profitability portfolios
29	Return on Assets	roa	Profitability	Operating Income Before Depreciation as a fraction of average Total Assets based on most recent two periods
30	Sales/Invested Capital	sale_invcap	Efficiency	Sales per dollar of Invested Capital
31	Small Minus Big	SMB	Other	Average return on the nine small stock portfolios minus the average return on the nine big stock portfolios
32	Labor Expenses/Sales	staff_sale	Other	Labor Expenses as a fraction of Sales
33	7-day Transaction Volume	Volume	Other	The 7-day rolling total transaction volume
34	Winner minus Loser	WML	Other	Equal-weight average of the returns for the two winner portfolios for a region minus the average of the returns for the two loser portfolios