Recurrent Networks for Factor Models in Time Series

Student: Tim Hung Wu (CID: 01195701), Supervisor: Dr Carlo Ciliberto, Second Marker: Professor Yannis Demiris. Course: EIE3

Abstract

This project explores the effectiveness of certain recurrent neural network architectures with long short-term memory units (LSTM RNNs) in the forecasting of stock prices for companies in the STOXX Europe 600 Industrial Goods and Services Index (SXNP). The project focuses on attempting to identify whether certain attributes of a company's fundamentals data, such as book value, significantly influences its stock price. An emphasis of this project was to implement a tool that would provide all the functionalities required to take this machine learning problem from the initial raw dataset to a reasonable final forecast result. As such, the various processes required to develop a solution to a machine learning problem such as data preparation, network structuring etc. were studied. By evaluating various hyperparameters and deepening the architecture, attempts were also made to improve the performance of the model. Finally, the returns that were predicted to be positive by the model were compared with general distribution of stock returns. The results show some capability of forecasting, with the model returning approximately 0.7 in accuracy, precision and recall metrics for the test set.

Background

Overview of Recurrent Neural Networks (RNNs)

- RNNs first introduced into literature by Rumelhart et al.[1] in 1986 to model cognitive perception.
- In contrast to feedforward networks like the multilayer perceptron, RNNs have a memory component. Its state is modelled as: $h_t = f_W(h_{t-1}, x_t)$
- Use cases: Sequential data tasks such as text generation, visual recognition and time series forecasting.

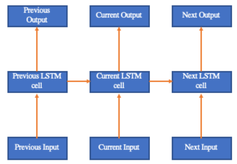

Figure 1: Unrolled architecture of RNN.

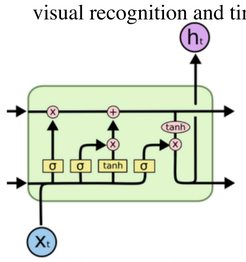

Figure 2: Bitwise logic of LSTM cell.

Vanishing Gradient Problem

- Repeated gradient signal multiplication across multiple cells during backpropagation causes small error gradients to tend towards 0 and vanish. [2]
- LSTMs were introduced by Hockreiter et al.[3] in 1997 to correct for this problem.
- LSTM units consist of three gates (forget, input and update) which process the flow of information through the cell and update its state.

Data Analysis

Data structure: Divided into 2 lists of features, *Fundamentals* and *Prices*.

- *Fundamentals*: General data about the company, such as book value. Subdivided into two categories: Last Twelve Months (LTM), the historical values for the last year, and Next Twelve Months (NTM), predictions for the next year.
- *Prices*: Data on the behavioural characteristics of the stock trades on the exchange.
- Total number of features for each category: LTM: 16, NTM: 31, *Prices*: 14.
- Each company is labelled with their unique ticker. There are 203 companies in the SXNP Index. Each feature contains 6313 time data points, from 3rd Jan 1995 onwards.

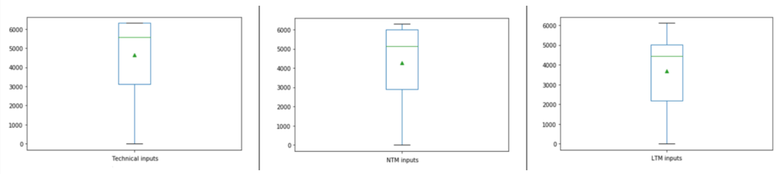

Figure 3: Boxplots indicating the approximate average underlying count numbers for each type of input. Green triangles indicate the mean counts.

Data sparsity problem: Solutions were considered about how to handle missing values.

- Three main methods: deleting, masking and generating values.
- Deletion: Resulted in a non standardised sequential input which caused very poor performance by the LSTM as the dataset lost its Markovian property.
- Masking: Caused a lot of repetitive data due to the low frequency of fundamental data updates. Produced extremely large datasets which were time consuming to train.
- Prediction: Required initially training on time series data with no missing values. Able to amend small gaps but compounding prediction errors in increasing data gaps.

Design and Implementation

The following figures describe the pipeline processes involved in setting up the ML experiment.

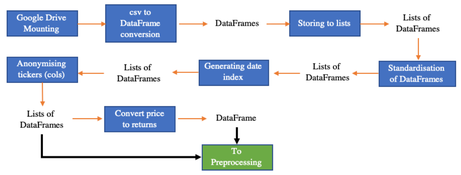

Figure 4: Program flow diagram of data structuring.

Step 1: Data Structuring

- *Objective: Convert raw datasets into manipulable data structures.*
- Data stored in Pandas DataFrames.
- Standardisation across all features.

Step 2: Data Preprocessing

- *Objective: Analyse the properties of the data and determine a suitable method of handling data for the algorithm.*
- Input attributes scaled between 0 and 1 using the min-max method.
- Target variable relabeled as binary values where +1 indicated positive return and 0 indicated negative return.
- Division into training, validation and test sets in the ratio of 60:20:20.

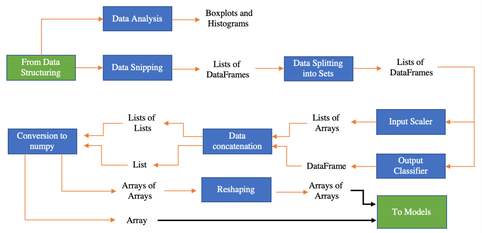

Figure 5: Program flow diagram of data preprocessing.

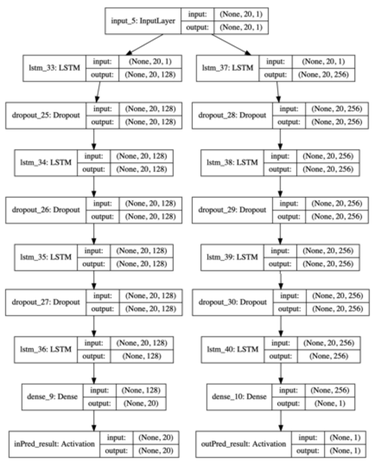

Figure 6: Architectural breakdown of multi output model.

Step 3: Neural Network Architecture

- *Objective: Design a learning algorithm that outperforms the naïve diversification benchmark.*
- Multi output architecture to provide predictions for both the next input and target variables.
- Inputs: 20×1 attributes vector. Outputs: 20×1 attributes vector and 1×1 binary digit.
- Stacked LSTM architecture for both branches consisting of 4 layers. Dropout rates of 20% used between layers to avoid overfitting.
- Deep architectures consistently outperformed shallow networks with the same total number of LSTM units.
- Loss functions: Binary cross-entropy and mean absolute error for output and input predictors respectively.
- Optimiser: Adaptive Moment Estimation (Adam).
- Batch size: 1024, Epochs: 30.

Step 4: Evaluation Metrics

- *Objective: Predict using the model and assess performance against the benchmark.*
- Selects all returns that are predicted to be +1 by the model in the test set and compares against the actual value of the returns.
- Visual plots and statistical summaries to compare against benchmark, which is all the returns of the entire test set.

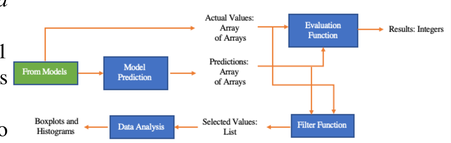

Figure 7: Program flow diagram of evaluation metrics.

References

Project GitHub Repository: https://github.com/twutang/SXNP-Factor-Models

[1] D. E. Rumelhart, P. Smolensky, J. L. McClelland, and G. E. Hinton. *Schemata and Sequential Thought Processes in PDP Models*. Morgan Kaufmann Publishers, 1986.

[2] Y. Bengio, P. Simard, and P. Frasconi. *Learning Long-Term Dependencies with Gradient Descent is Difficult*. IEEE Transactions on Neural Networks, 1994.

[3] S. Hockreiter and J. Schmidhuber. Long Short Term Memory. MIT Press, 1997.

Results

- Model predictions exhibited more preferable statistical properties than the test set benchmark. Accuracy plateaus at ~0.7 for training, with the same result in test data.
- Further hyperparameter optimisations such as decayed learning rate or alternate weight initisalisation may have improved performance.

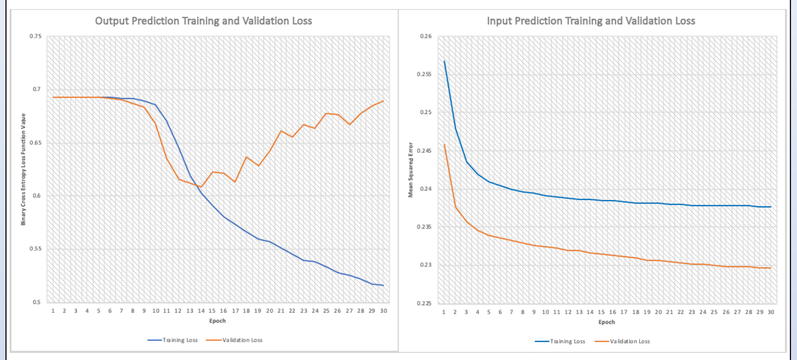

Figure 8: Training and Validation losses for next input and target variable prediction by multi output model.

Accuracy	Precision	Recall
0.705	0.700	0.699

Table 1: Model performance metrics for holdout test data.

	Entire	Test	Selected
Mean	0.000389	-0.000147	0.00764
Variance	0.000546	0.000364	0.000269
Skew	64.114	-1.489	1.224
Kurtosis	18474.2	43.882	31.110

Table 2: Summary of returns data distributions for the entire, test and model predicted datasets.

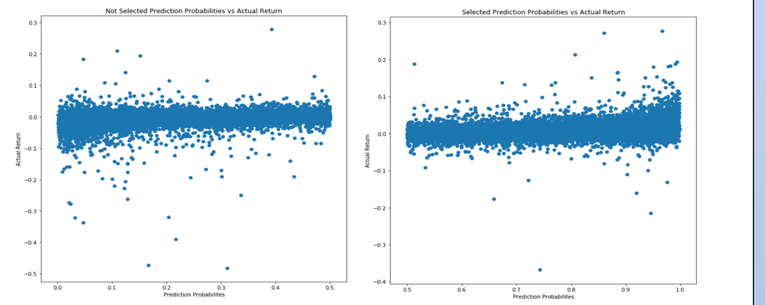

Figure 9: Scatter plots of the unselected (<0.5) and selected (>0.5) prediction probabilities against actual returns of the datapoint.

Conclusion

The project effectively evaluated a theoretical application of recurrent networks to the problem of predicting prices. One of the most significant achievements of this project was the implementation of a model which showed signs of outperformance in terms of prediction against the designated benchmark. Although the input data did not solely rely on fundamental data as initially desired, the addition of technical analysis data was deemed vital for the success of the model.

However, it would be hard to translate any of the research generated into a practical investment application. For example, one of the issues arising from the problem is that the model takes the company metrics and predicts the price at that specific time step t. This does not translate into a viable long term investment plan as the objective in that is to evaluate the company metrics and judge the future time at a time step $t + c$ where c is the investment horizon period.

Future Work

- Data Feeds: Add more types of data to see whether this would add any improvement. For example, sentiment data, such as newsfeeds could be useful.
- Alternate learning algorithms: Experiment with a reinforcement learning approach. An example framework would be to use the profit and loss of a portfolio as the reward, with buying and selling stocks being the decision and action processes.