Stock Price Prediction with LSTM - Draft

Roger Bukuru (BKRROG001)

University of Cape Town
Intelligent Systems (CSC5025Z)
Draft Report

1 Experimental Design

In this study we applied the Recurrent Neural Network (RNN) method known as Long-Short-Term Memory to forecast prices for 30 FTSE/JSE Top 40 Index constituent shares from the year 2012 to 2021, the data represented the daily closing prices of these stocks over this period. To perform the forecast, we built a machine learning pipeline (figure 1), that achieve the following:

1. Reads the data received via a csv file.
2. Scales the data using using Z-score normalisation, where we remove the mean and re-scale to unit variance
3. Feature engineering by using sliding windows of either 30, 60, or 120 time steps and future time horizons of 1,2, 5, 10, 30 time steps. This result in new data set
4. Utilizes the created data set from (3) to split the data into training, validation and set according to the following proportion (70%, 15%, 15%)
5. Performs training using a hyperparameter grid for each of the (sliding window, horizon) combination to find the optimal model architecture. Resulting in the optimal model parameters based on the parameter that minimizes loss on the validation set.
6. The optimal parameters for each model are then used and evaluated against the test data set. This is executed across three runs to ensure model stability.
7. The model with the optimal performance on the test data is saved and will be used for inference.

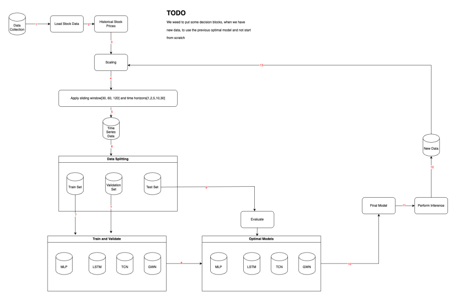


Fig. 1: ML Pipeline

Our LSTM included the following components:

- Input Layer: These are the time steps that was dependent on the window size(30, 60, 120)
- Hidden Layers: We explored two different dimension sizes, discussed in the next section.
- Output Layer: Given we are doing single stock prediction our output size represented the number future time steps i.e the horizons (1,2,5,10,30)
- Dropout Layer: We applied two different drop out rates
- Hidden Cell States: Initialized to zero at start of each input sequence, serving as the networks memory
- Optimizer: We utilized the Adam optimizer with a fixed learning rate of 0.001

2 Hyperparameter Optimisation

The hyperparameter optimisation was undertaken across 50 epochs for each of the following model configurations and hyperparameter space as shown in 1 below:

Window Size (days)	Prediction Horizon (days)
30	1, 2, 5, 10, 30
60	1, 2, 5, 10, 30
120	1, 2, 5, 10, 30

Tab. 1: Combinations of Window Sizes and Prediction Horizons for Stock Price Prediction Models

For each model combination (15 models), the following hyperparameter space shown in 2 was explored.

Tab. 2: Hyperparameter Space Explored for Each Model Combination

Hyperparameter	**Values**
Hidden Dimension	32, 64
Number of Layers	2, 8
Dropout Rate	0.1, 0.2
Learning Rate	1×10^{-3}
Batch Size	32, 64

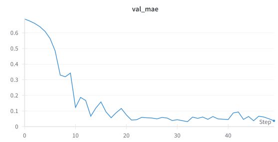
Note: For each combination of window size ($W \in \{30, 60, 120\}$ days) and prediction horizon ($H \in \{1, 2, 5, 10, 30\}$ days), the hyperparameters are explored across all possible combinations of the specified values for the following models: MLP, LSTM, TCN (single share prediction) and GraphWaveNet (multi-share prediction). This results in a total of $2 \times 2 \times 2 \times 1 \times 2 = 16$ hyperparameter configurations per window size and horizon combination, leading to an overall exploration of $3 \times 5 \times 16 = 240$ models.

2.1 Optimal Model Configuration

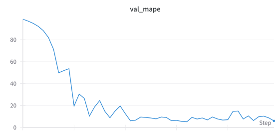
After conducting the hyperparameter optimisation, in table 3 we demonstrate the optimal hyperparameters for each model configuration, we note the metrics were not denormalized. For each configuration we highlight in bold the optimal model for each configuration and show in figure x their training convergence over the validation metrics.

Configuration		Batch Size	Dropout Rate	Hidden Dim	Num Layers	Learning Rate	Output Dim	Train MAE	Train MAPE	Train RMSE	Val MAE	Val MAPE	Val RMSE	Train Time (s)
30	**1**	**32**	**0.2**	**64**	**2**	**0.001**	**1**	**0.0904**	**10.1973**	**0.1092**	**0.0372**	**6.0628**	**0.0458**	**5.274**
	2	64	0.1	64	2	0.001	2	0.0896	10.8422	0.1161	0.0441	7.4123	0.0575	7.744
	5	32	0.1	64	2	0.001	5	0.1189	27.1667	0.1485	0.1508	131.6888	0.2455	2.127
	10	32	0.2	32	2	0.001	10	0.1786	111.9552	0.2328	0.1715	47.0056	0.3022	3.676
	30	32	0.2	64	2	0.001	30	0.2099	23.5627	0.2660	0.2209	42.4887	0.3646	5.301
60	**1**	**32**	**0.1**	**64**	**2**	**0.001**	**1**	**0.0790**	**8.6942**	**0.1191**	**0.0343**	**6.5300**	**0.0460**	**1.505**
	2	32	0.2	64	2	0.001	2	0.1226	12.3340	0.1790	0.0769	13.6360	0.0973	3.701
	5	64	0.2	64	2	0.001	5	0.2054	34.8338	0.2547	0.1017	18.9479	0.1327	6.627
	10	32	0.2	64	2	0.001	10	0.1329	17.6982	0.1901	0.0479	8.5290	0.0681	3.745
	30	32	0.1	64	2	0.001	30	0.1915	19.2491	0.2688	0.1224	17.9480	0.1585	1.537
120	1	32	0.1	64	2	0.001	1	0.1073	10.9471	0.1610	0.0997	18.2230	0.1240	0.927
	2	**64**	**0.2**	**32**	**2**	**0.001**	**2**	**0.3193**	**34.9080**	**0.3889**	**0.0563**	**10.0242**	**0.0594**	**4.489**
	5	64	0.1	64	2	0.001	5	0.1875	24.9092	0.2287	0.1398	30.4330	0.1792	3.758
	10	32	0.1	64	2	0.001	10	0.2686	39.5272	0.3513	0.1057	20.4421	0.1473	2.338
	30	32	0.2	64	2	0.001	30	0.3018	42.9030	0.3949	0.1343	16.4684	0.1633	2.352

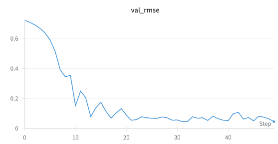
Tab. 3: Optimal Model Configurations for Each Window Size and Prediction Horizon with Training Metrics



(a) Validation MAE

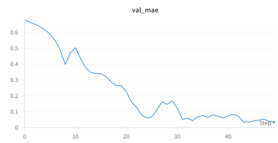


(b) Validation MAPE

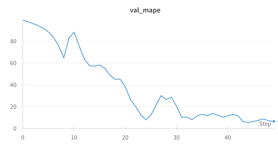


(c) Validation RMSE

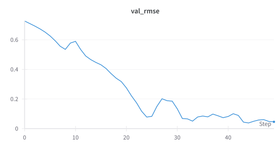
Fig. 2: Window 30 Horizon 1 Optimal Model



(a) Validation MAE

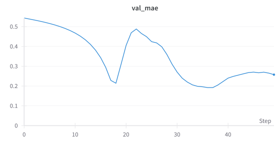


(b) Validation MAPE

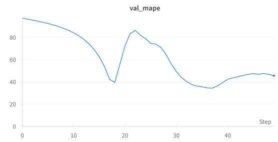


(c) Validation RMSE

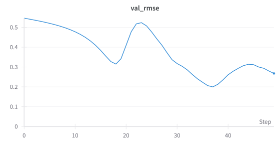
Fig. 3: Window 60 Horizon 1 Optimal Model



(a) Validation MAE



(b) Validation MAPE



(c) Validation RMSE

Fig. 4: Window 120 Horizon 2 Optimal Model

We note that no optimal model had better performance than a prediction horizon of 1-2 days, which aligns with our intuitive understanding that further timestamps introduces complexity and extensive additional training would be required to get an optimal model beyond a 2 day horizon. Moreover we not that both the 30 window and 60 window optimal models had near exact performance when we analyse the their validation metrics, this indicates that we could opt for the simpler 30 window model for 1 time-step(day) forecasting and the 120 window model for 2 time-step(day) forecasting.

2.2 Evaluation on Test Data

We executed each model on the test data set across 5 runs for model stability, in figure 5 below we show the our results.

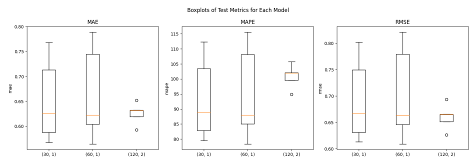


Fig. 5: Model Stability

We observe that although the (60,1) model had the lowest loss across the different metrics, it had the highest variation around the mean across the 5 runs. The (120,2) model had the lowest variability, however we do note a significant fluctuation in the mean when we observe it's MAPE performance. This leave the (30,1) model the most consistent across all three metrics despite have more variability then the (120,2) model. We would therefore recommend utilizing the (30,1) model for it's consistency and stability.