Image-based Stock Return Prediction with CNN

MAFS 6010Z Project 2

Wang Lei, Wang Zhongchen, Ye Xiaoyu and Zhang Quandi

{lwangcu, zwangfz, xyeak, qzhangcd}@connect.ust.hk

Department of Mathematics, HKUST

1. Introduction

This report summarizes the replication study on paper '(Re-)Imag(in)ing Price Trends' (hereinafter called 'the paper'), which innovatively applies CNN to image-based stock return prediction. To be more concise and focus on the main idea, and to avoid repetitive work, we choose the medium(20-day) horizon to replicate. We construct the 3 core building blocks of the CNN model according to the instructions in the paper and do hyperparameter tuning works based on the results on the validation set for those not covered in the paper.

We follow the appendix in the paper to evaluate the performance of the CNN image classifier by comparing the accuracy, correlations and Sharpe ratios of variation of hyperparameters and visualize the returns of long-short portfolios.

2. Replication

2.1 Data Preparation

We define the class `ImageData` for data preparation. The dataset is split into train, validation and test dataset inconsistent with the settings in the paper: Firstly, seven-year samples (1993-1999) as training and validation dataset, in which 70% is randomly selected for training and the remaining 30% as validation; 20-years samples (2000-2020) as test dataset (Jiang, Kelly & Xiu, 2020).

The followings are some critical steps when processing the data. Firstly, all image labels with a value of 1 or 0 for positive or non-positive returns in the following 20-day period are extracted from the whole dataset. *NaNs* are removed by filtering out those with a label value of 2. Secondly, since the input images are in RGB format whose pixel values range from 0 to 255, the *PyToch* `transformer` is applied to convert and normalize the data to tensor format valued from 0 to 1. Finally, `__getitem__` is used to return a pair of image data and its label by assigning an index to match the corresponding date and stock ticker, and thus facilitating later calculation of performance evaluation indicators. Specifically, in the testing process, we save all predicted labels `pred_label` corresponding to their testing image data, and the index here acts as a bridge connecting the testing data and the result. In that case, we add a new column "pred_label" storing all predicted outputs and using them for calculations.

2.2 Model Design and Workflow

Most of the parameters of baseline model are consistent with the paper, except the followings which are not mentioned or need appropriate adjustments.

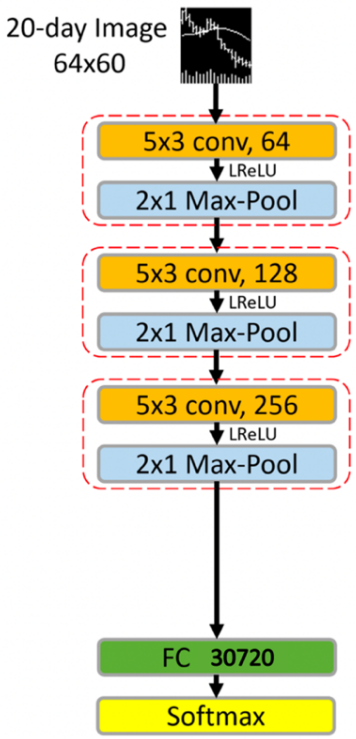
A larger *batch size* of 1280 for training and 3000 for testing is used (instead of 128 in the paper (Jiang, Kelly & Xiu, 2020)) to speed up the training process. Correspondingly, the *learning rate* of 0.0001 is used as the *learning rate* should be scaled up by the same multiplier (Krizhevsky, 2014). The

multiplicative factor γ of learning rate decay is set to be 0.9, which is not mentioned in the paper, and it is the best setting according to the tuning result on the validation dataset. Moreover, based on the early stop principle of loss function failing to improve for two consecutive epochs, we finally adopt 14 epochs for the sake of the largest efficiency and accuracy.

We notice a mismatch of model hyperparameter stride = (3,1) and the output size mentioned in the paper (46,080 for 20-day data), and the actual output size should be 30,720. If we want to maintain the output in paper which is 46,080, the strides need to be modified as (2,1). To maintain consistency, we stick to the strides of (3,1). After the 3 convolutional blocks, the baseline model has connected a fully connected layer with an input size 30,720 and drop out of 0.5 to avoid overfitting (Jiang, Kelly & Xiu, 2020).

The final architecture of the baseline model is as the figure shows below.

Figure 1: Diagram of CNN model



2.3 Performance Evaluation

On the validation set, the baseline model has a 0.690 cross-entropy loss (0.687 in paper) and a 54.1% accuracy (54.2% in paper) (Jiang, Kelly & Xiu, 2020). On the test set, the cross-entropy loss is 0.698 (0.690 in paper) and the accuracy is 51.9% (53.3% in paper). Our results are close to the results in paper to a great extent.

Table 1: Performance Comparison

	Loss		Acc.	
	V	T	V	T
Baseline (Replication)	**0.690**	**0.698**	**0.541**	**0.519**
Baseline (Paper)	0.687	0,690	0.542	0.533

3. Sensitivity Analysis

3.1 Model Structure and Indicators

We follow the appendix to train 16 more models for sensitivity analysis. Each of them has variation of one hyperparameter only and the structure of variant models follow the Table 18 of the paper (Jiang, Kelly & Xiu, 2020), including the number of filters in the first layer, the number of convolution layers, the dropout rate, use of batch normalization, Xavier initialization, different activation functions, size of Max pool, filter size, and the dilation/stride. The table, which is written in brackets means the parameters of baseline. Also, we adopt similar indicators (loss and accuracy, Spearman and Pearson correlation of validation and test set; annualized Sharpe ratios with the different risk-free rates of 0% and 1% of test set only), to evaluate and compare the performance and indicate the impact of each hyperparameter.

3.2 Calculation of Indicators

For the correlations, we first collect all the realized predicted labels throughout the testing period (2000-2020) and calculate the correlations based on two methods, Pearson and Spearman, and apply `scipy.stats.pearsonr` and `scipy.stats.spearmanr` to calculate them.

As the paper does not mention the details of long-short decile strategies and the portfolio construction for calculating Sharpe ratio, we adopt the equally-weighted method, which means we allocate the same absolute weightings for every stock in the data and give a positive (long position) or negative (short position) sign based on the predicted label, 1 for '+' and 0 for '-'. More specifically, for n stocks in the stock pool, the weighting of stock for the next 20 days is $w = 1/n$ when the predicted label is 1 and $w = -1/n$ when the predicted label is 0. And we rebalance the portfolio when the next predicted label occurs. The financial meaning of the same absolute weightings of long and short is that the margin of short selling is 100%, which is a very conservative strategy.

After collecting all portfolio returns on each 20-day period, we calculate Sharpe ratio of the 20-day horizon:

$$SR_{20day} = \frac{Average(\ returns_{20day}) - rf_{20day}}{\sigma_{20day}}$$

Then annualize the Sharpe ratio by (252 trading days per year):

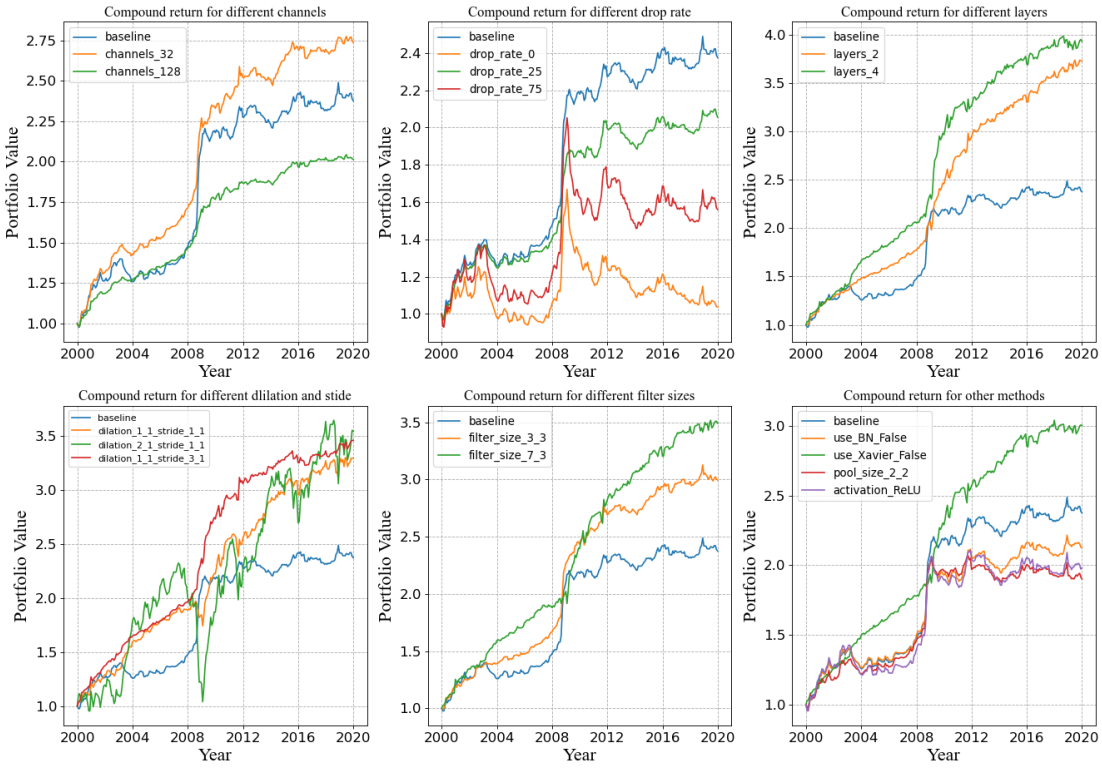
$$SR_{annual} = SR_{20} \times \sqrt{\frac{252}{20}}$$

As the table below, we report loss function value and classification accuracy (both the validation (V) and test (T) samples), cross-sectional correlation (T) (both Pearson and Spearman rank), and annualized Sharpe ratios for risk-free rate rf=0% and rf=1%. Figure 2 shows the values of portfolios change over time.

Table 2: Sensitivity to Model Structure and Estimation

		Loss		Acc.		Correlation		Sharpe Ratio	
		V	T	V	T	Spearman	Pearson	Rf=0	Rf=.01
Baseline		**0.690**	**0.698**	**0.541**	**0.519**	**0.029**	**0.023**	**0.808**	**0.637**
Filters(64)	32	0.688	0.694	0.540	0.520	0.028	0.022	0.965	0.774
	128	0.699	0.709	0.534	0.515	0.016	0.009	1.009	0.791
Layers(3)	2	0.689	0.696	0.540	0.522	0.027	0.019	1.129	0.913
	4	0.691	0.697	0.536	0.522	0.025	0.017	1.195	0.983
Dropout(0.50)	0.00	0.719	0.733	0.531	0.509	0.016	0.011	0.853	0.675
	0.25	0.695	0.704	0.538	0.515	0.020	0.015	0.852	0.669
	0.75	0.689	0.695	0.539	0.517	0.033	0.028	0.723	0.560
BN (yes)	No	0.689	0.696	0.542	0.517	0.026	0.020	0.723	0.560
Xavier (yes)	No	0.694	0.702	0.537	0.518	0.019	0.012	0.768	0.599
Activation (LReLU))	ReLU	0.689	0.697	0.542	0.518	0.029	0.024	0.746	0.580
Max-pool Size (2×1）	(2×2）	0.690	0.697	0.535	0.517	0.025	0.019	0.745	0.568
FilterSize (5×3)	(3×3）	0.688	0.695	0.541	0.522	0.031	0.025	0.765	0.596
	(7×3）	0.691	0.698	0.536	0.519	0.021	0.013	0.798	0.627
Dilation/Stride (2,1)/(3,1)	(2,1)/(1,1)	0.725	0.731	0.510	0.514	-0.011	-0.030	0.764	0.552
	(1,1)/(3,1)	0.691	0.697	0.537	0.520	0.024	0.017	0.723	0.578
	(1,1)/(1,1)	0.699	0.703	0.530	0.518	0.016	0.008	0.740	0.594

Figure 2: Visualization of Portfolio Returns



3.3 Key Findings

By comparing the loss and accuracy on validation and test, a stable but acceptable difference is observed, which indicates a potential overfitting problem on the validation dataset. The differences between variant models are small, suggesting that no single hyperparameter plays a significant role in the classification. In terms of correlations coefficients, all of them are small, and even some of them are negative (no matter in our replication or in the paper), indicating a poor classification performance or there is no linear relationship.

We notice that when the number of filters falls from the baseline filter (64) to 32, the test accuracy is slightly improved, while if it rises to 128, both the accuracy and correlation have worse performance. For layers, when the number of layer transfer to 2 or 4, the result of correlation slightly grows but the Sharpe ratios rapidly decline. The accuracy on the test set increases with the increase of the dropout rate, indicating the effectiveness of avoiding overfitting.

Transferring the BN from yes to no would reduce the performance of the model. Changing the Xavier would basically have no effect on the performance, which means the initialization is not a key factor for the optimization. The model performance is insensitive to the choice of activation function because when we change it from LReLU to ReLU, the result stays similar. Any listed adjustments of Max pooling size and filters' size have a negative impact on the model performance. Lastly, we can see the

performance be slightly affected when we modify the dilation. However, if we change the stride, the performance will substantially drop.

Figure 2 summarizes the portfolio returns and provides an intuitive cross-model comparison. From the plots, we can see that all portfolios reflect the fluctuation of the 2008 financial crisis and show disparity after the crisis (especially the plot of drop rate). Some of the portfolios perform worse in the post-crisis period, indicating weaker prediction power and low adaptation of the models after a huge change in the external financial environment.

In conclusion, the change of layers has the most significant impact on the model performance and also give best returns over time, and the baseline is given by the paper still has the space for improvement.

4. Self-reflection and Future Work

There are some interesting findings in the replication process. First, when the learning rate is too small, the use of Xavier makes the convergency process more difficult and slower. The reason may be that learning from the initialization of Xavier with a small learning rate makes it harder to go beyond a local minimum rather than the global minimum we are looking for. Second, when the number of epochs is large, models tend to be overfitted to the training dataset. A large portion of dropout can be applied to improve the results.

As the perdition power of the CNN in the paper is relatively weak, we can consider modifying the structure of the building blocks to find a better method. More tuning can be done based on the results of the test set to avoid potential overfitting problems.

5. Conclusion

The work of the paper is replicable, and our results of loss, accuracy are close to the original results while the correlations of our replication are lower. The reason could be the difference in calculation methods and/or the random seeds. However, with the absence of a portfolio construction method (the long-short decile strategies), the Sharpe ratios are not verifiable, and our results are not ideal. Furthermore, the paper gives a classification accuracy of 53% of the baseline model on the test dataset indicate a very low prediction power no matter how good the Sharpe ratio is because the main driven factor of the profitability could be the strategy of portfolio construction instead of the classification. In conclusion, although the model results are replicable, and the idea is instructive, we still consider the practicality of the paper is low.

References

Jiang, J., Kelly, B., & Xiu, D. (2020). (Re-)Imag(in)ing Price Trends. *SSRN Electronic Journal*. doi: 10.2139/ssrn.3756587

Krizhevsky, A. (2014). One weird trick for parallelizing convolutional neural networks. *ArXiv, abs/1404.5997*.

Contribution:

Coding:

Model construction and model training: YE Xiaoyu

Preparation of return data: WANG Lei

Plot: WANG Zhongchen

Report writing:

Intro& Conclusion: WONG Zhongchen

Replication: WONG Lei

Sensitivity analysis: ZHANG Quandi

Self-reflection and future work: YE Xiaoyu