Reproduction: (Re-)Imag(in)ing Price Trends

Xinzhen Liao
Yuan Xu
Rui MA
Yiyuan Shi

Nov. 14, 2021

CONTENTS

LIST OF FIGURES

Reproduction:
(Re-)Imag(in)ing Price Trends

***Abstract*—Based on the article (Re-) Imag(in)ing Price Trends, we consider using methods that flexibly learn price patterns which are most predictive of future returns to predict future returns. Most importantly, the raw predictor data we used are not conventional numeric data, but images from which authors model the predictive association between images and future returns based on the Deep Learning algorithm – convolution neural network (CNN). According to the article (Re-) Imag(in)ing Price Trends, it explains that CNN is able to automatically identify context-independent predictive patterns which could give more accurate return predictions, translate into more profitable investment strategies and robust to variations.**

This whole report will be divided into 6 separate parts, which are Background and Introduction, Data, Model Design, Workflow, and Prediction and Interpretation.

I. Introduction

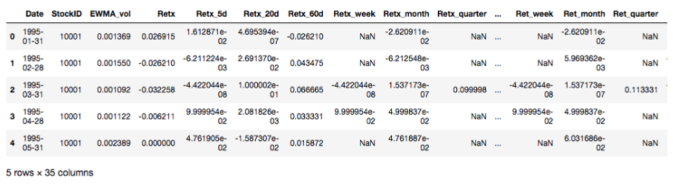
Globally, stock market is one of the most emerging and important sectors. It becomes essential and vital to know about the market trends for all people in the near decade, no matter for retail investors or professional financial origination. As a result, with the solid development of the stock market, people now become more interested in forecasting the stock price. However, due to the complicated, subtle, and dynamic nature of stock market, following with the quick changes in stock price, it is really challenging to accurately forecast the stock price.

Nowadays, lots of researchers have already investigated and applied plenty of methods to predict stocks' future price with the help of techniques like Machine Learning and Deep Learning. Momentum, Mean Reversion, and Martingales are some common methods that widely applied today to forecast the stock price. Unlike traditional methods, in this report, we will reconsider the idea of price-based return predictability from a different perspective, the "Image" Market Data. [1]

II. Data

A. The Market Data

According to the article (Re-) Imag(in)ing Price Trends, the empirical analysis revolves is a panel prediction model for US stock return from 1993 to 2019. And the data input to this model are images depicting price and volume (daily open, close, high, and low prices, daily trading volume, and moving average price) over the past 5, 20, and 60 days. While the output of the CNN is a set of stock-level estimates for the probability of a positive subsequent return over shot (5-day), medium (20-day), and long (60-day) horizons. Part of the original data we used to calculate our prediction's accuracy in further part is shown in figure 1.



	Date	StockID	EWMA_vol	Retx	Retx_5d	Retx_20d	Retx_60d	Retx_week	Retx_month	Retx_quarter	...	Ret_week	Ret_month	Ret_quarter
0	1995-01-31	10001	0.001369	0.026915	1.612871e-02	4.695394e-07	-0.026210	NaN	-2.620911e-02	NaN	...	NaN	-2.620911e-02	NaN
1	1995-02-28	10001	0.001550	-0.026210	-6.211224e-03	2.691370e-02	0.043475	NaN	-6.212548e-03	NaN	...	NaN	5.969362e-03	NaN
2	1995-03-31	10001	0.001092	-0.032258	-4.422044e-08	1.000002e-01	0.066665	-4.422044e-08	1.537173e-07	0.099998	...	-4.422044e-08	1.537173e-07	0.113331
3	1995-04-28	10001	0.001122	-0.006211	9.999954e-02	2.081826e-03	0.033331	9.999954e-02	4.999837e-02	NaN	...	9.999954e-02	4.999837e-02	NaN
4	1995-05-31	10001	0.002389	0.000000	4.761905e-02	-1.587307e-02	0.015872	NaN	4.761887e-02	NaN	...	NaN	6.031686e-02	NaN

5 rows × 35 columns

Fig. 1. Original Data

Here is part of the detailed explanation for each label.

- **Date:** the last day of the 20-day rolling window for the chart
- **Stock ID:** CRSP PREMNO that identifies the stock
- **Market Cap:** Market capitalization in dollar, recorded in thousands
- **Ret_td:** t=5,20,60, next t-day holding period return
- **Ret_month:** Holding period return for the next month, from the current month end to the next month end
- **EWMA_vol:** exponentially weighted volatility (square of daily returns) with alpha 0.05. One day delay is included

We divided the original data-set into 3 categories based on the returns. Where 0 stands for non-positive returns (i.e., the down trend), 1 stands for positive returns (i.e., the uptrend), and 2 stands for the situation contains a NaN result.

B. The "Image" Data

As introduced above, the original data contains three scales of horizons, which are short horizon (5-day), medium horizon (20-day), and long horizon (60-day). For the long horizon (60-days), the network structure will be more complicated with a lower computing efficiency. Also, the short horizon (5-days) is too simple, which might not fail to draw a solid conclusion. Thus, in this report, we only use the medium horizon instead of all of them.

The current images have the same resolution (64×60) and added with moving average lines (MA) with the volume bars (VB). Besides, we standardize all the "Image" data into the binary form, cutting down the data size from 255 to 1.

III. Model Design

A Convolution Neural Network (CNN) is a Deep Learning algorithm which can take in an input image, assign importance to various objects in the image and be able to differentiate one from the other. Also, a CNN is a modeling scheme that stacks together a sequence of operations to transform a raw image into a set of predictive features following with a prediction at the end. Convolution, Activation, and Pooling are three key operations of a building block.

A. Convolution

The convolution operation uses a set of “filters.” A filter is a low dimension kernel weighting matrix. In this project, we used 64 filters with a 5×3 kernel size. The short horizon, medium horizon, and long horizon CNN models are built with 2, 3, and 4 CNN building blocks respectively. The detailed diagram for the 20-day situation is shown in figure 2.

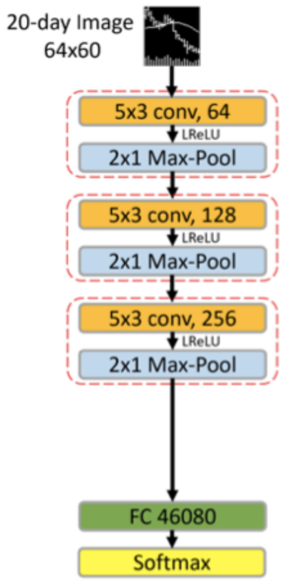


Fig. 2. 20-days Image Convolution

As for this part, we add the padding function to protect data from missing. The function padding increases the number of pixels on each side to prevent feature map from becoming too small. The specific increase in the number of pixels is determined by the size of filter and stride.

In addition, we use batch normalization in our CNN model design to prevent model from over-fitting. Batch normalization is a layer that allows every layer of the networks to do learning more independently and it is also used to normalize the output of the previous layers. It is very efficient in CNN and used to prevent the model from over-fitting through the regularization.

B. Activation

Activation is the second operation in a building block. This operation is a linear transformation applied element-wise to the output of a convolution filter. And the activation function we use in this report is the “leaky ReLU”.

C. Pooling

Pooling is the third operation in a building block. This operation is conducted for 2 main reasons. Firstly, max-pooling could reduce the dimension by getting rid of redundant information. Secondly, max-pooling is used to enhance CNN robustness as a de-noising tool.

IV. Working Flow

A. Data Split

Our group follows the sample classification interval classification method in the original paper, and we divide the samples into training, validation and testing samples. The seven-year samples from 1993-1999 are used to train and validate the model, in which 70% of the randomly selected samples are used for training, and the remaining 30% are used for verification. The remaining data from 2000 to 2020 includes out-of-sample test data sets.

B. Train Process

For 20-day images, we following the paper to build a baseline CNN model, the details have been introduced above. In order to combat over fit and aid efficient computation, we apply same regularization procedures, including the Xavier initialization, dropout, batch normalization and early-stopping.

1) the Xavier Initialization: We apply the Xavier initializer for weights in each layer. With the passing of each layer, the Xavier initialization maintains the variance in some bounds so that we can take full advantage of the activation functions. This innovative method achieves faster convergence by generating the initial value of the weight, thereby ensuring that the prediction variance starts at a comparable scale of the label. From our experimental results, the Xavier initialization has greatly improved the convergence speed of our CNN model, and has also improved the classification and prediction effects.

2) Dropout: During training, we apply 50% dropout to the fully connected layer, re-scaling the weights of neurons that have not been dropped out. Dropout is a method used by neural networks to prevent over-fitting. It is to give up activated neurons with a certain probability, which is equivalent to giving up some Features, which makes the model not overly dependent on certain features, even if these features are real. Dropout greatly improves the stability and robustness of our model.

3) Batch Normalization: We use batch normalization in our CNN model design to prevent model from over-fitting. Batch normalization is a layer that allows every layer of the networks to do learning more independently and it is also used to normalize the output of the previous layers. It is very efficient in CNN and used to prevent the model from over-fitting through the regularization.

4) Early-Stopping: We also set the early-stopping by calculating the performance of the model on the validation set during training. When increasing the number of epochs no longer significantly improves the performance of the model on the validation set or even has a decline, we will stop the training process so as to avoid the problem of over-fitting caused by continued training.

C. Evaluation

1) Cross Entropy Loss: We simply treat the prediction analysis as a classification problem. And use Cross Entropy Loss as our loss function to measure the performance of our model:

$$L_{CE}(y,\hat{y}) = -ylog(\hat{y}) - (1-y)log(1-\hat{y})$$

2) Accuracy: We use Accuracy to measure the classification accuracy, which represents the ratio of the number of correct determinations to all the number of determinations. The number of correct judgments is $(TP + TN)$, and the number of all judgments is $(TP + TN + FP + FN)$, so the expression of accuracy is as follows:

$$Accuracy = (TP + TN)/(TP + TN + FP + FN)$$

3) Classification Performance: After applying same regularization procedures, including the Xavier initialization, dropout, batch normalization and early-stopping, the classification performances of our model is shown in figure 3 below:

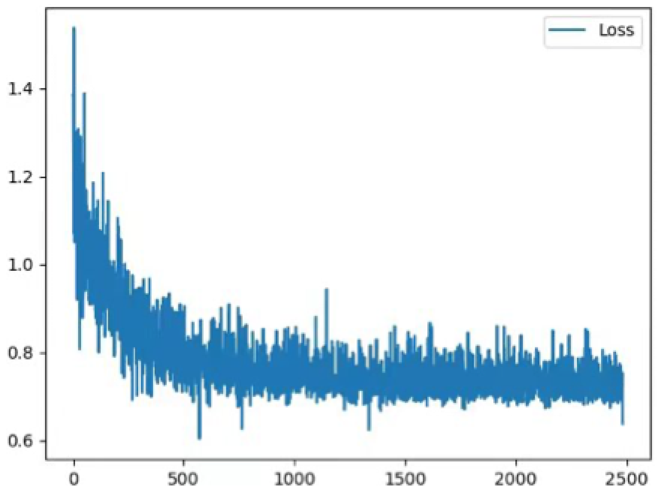


Fig. 3. Cross Entropy Loss Value VS Number of Iterations

The above figure shows the relationship between cross entropy loss value and the number of iterations. As the iterations increase, the loss value of our classification model has a relatively obvious trend of convergence.

Figure 4 & 5 shows the relationship between cross entropy loss value and accuracy changes with the number of epochs. For our validation and test data set, the number of epochs stopped early when it reaches about 12. In terms of classification effects, the performance of train and validation data-set is not much different. The loss value is approximately stable at about 0.7, and the accuracy is approximately stable between 0.51-0.53. For our test data-set, the loss value is approximately 0.7375, and the accuracy is 0.5082, which is closer to the result in the paper.

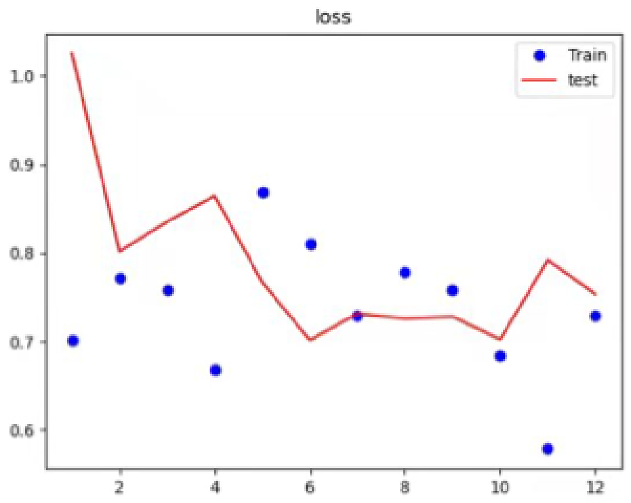


Fig. 4. Cross Entropy Loss VS Number of Epochs

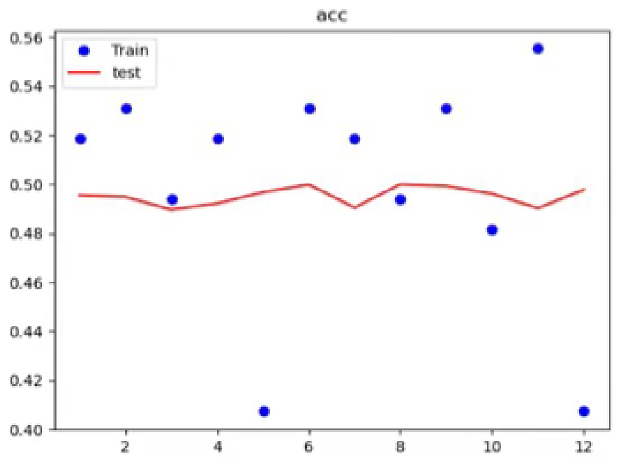


Fig. 5. Accuracy Changes VS Number of Epochs

V. Extensions

A. Ablation studies and test robustness

We also try to change some parameters or some features of model to test the robustness, for example, varying the dropout rate(0.25, 0.5), learning rate of stochastic gradient descent($1 \times 10^{-5}, 1 \times 10^{-3}$), batch size(32, 64, 128, 256).

1) Dropout Rate: When we keep learning rate at 1×10^{-3}, batch size at 128, without dropout and the Xavier initialization(Fig. 6), there is an obvious over-fit phenomenon, the loss value of validation data-set is much lower than train data-set. [2]

When we keep learning rate at 1×10^{-3}, batch size at 128(Fig. 7), dropout rate at 50% and without the Xavier initialization, there is a significant improvement in over-fitting, there is little difference between the loss value of validation data-set and that of train data-set.

When we want to analysis the impact of changing the

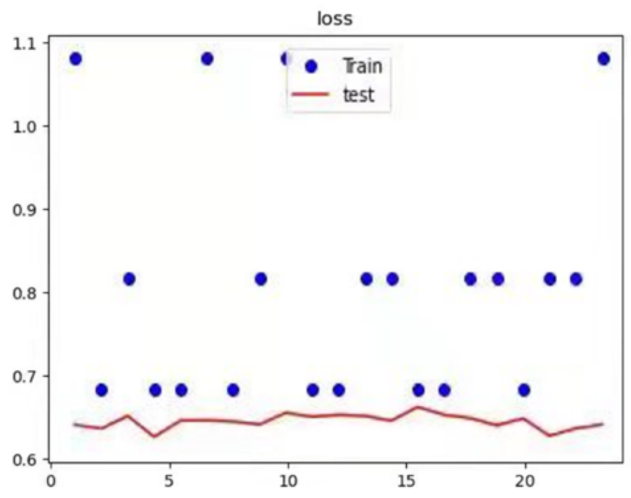


Fig. 6. Learning Rate = 1×10^{-3}, Batch Size = 128

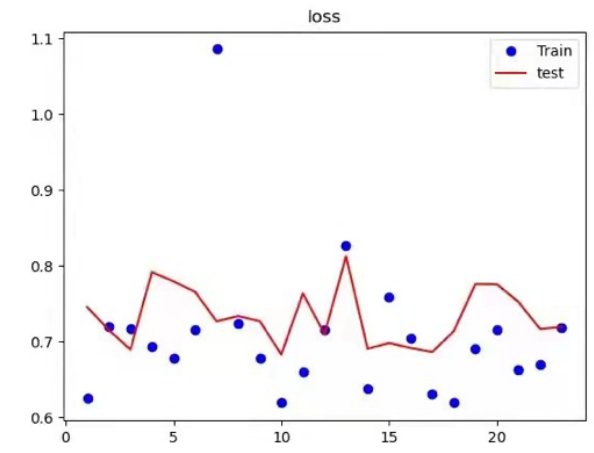


Fig. 7. Learning Rate = 1×10^{-3}, Batch size = 128, Dropout Rate at 0.5

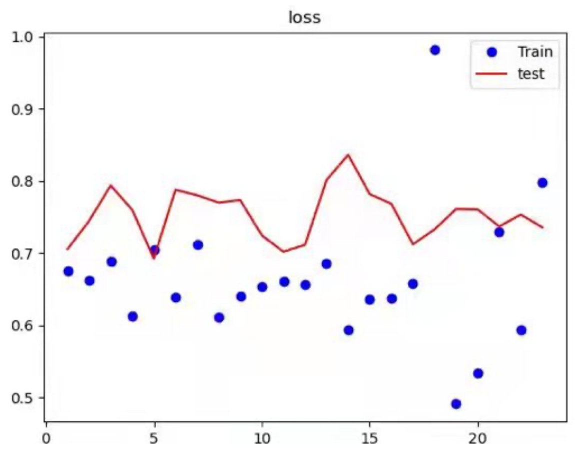


Fig. 8. Learning Rate = 1×10^{-3}, Batch size = 128, Dropout Rate at 0.25

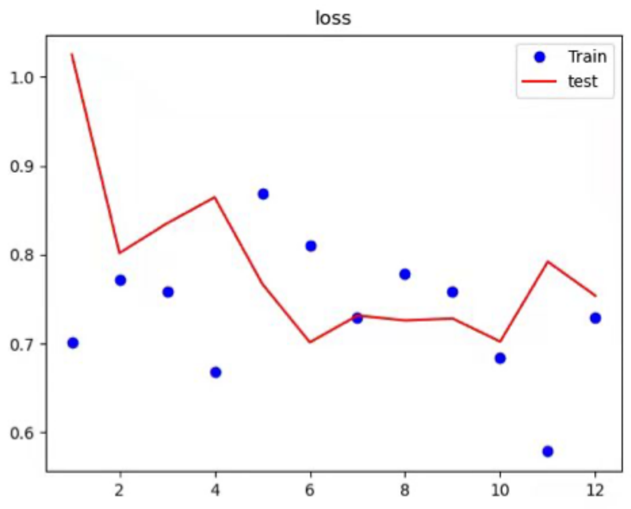


Fig. 9. Learning Rate = 1×10^{-3}

dropout rate on the robustness of the model, we still keep learning rate at 1×10^{-3}, batch size at 128, and without the Xavier initialization, but we set dropout rate at 25%, there is no significant improvement in the performance of our model(Fig. 8).

2) Learning Rate: When we want to analysis the impact of changing the learning rate on the robustness of the model, we keep dropout rate at 50%, batch size at 128, and with the Xavier initialization, but we first set learning rate at 1×10^{-3}, the loss value is approximately stable between 0.7-0.8(Fig. 9).

When we keep dropout rate at 50%, batch size at 128, and with the Xavier initialization, but this time we set learning rate at 1×10^{-5}, there is no significant improvement in the performance of our model(Fig. 10).

3) Batch Size: We also analysis the impact of changing the batch size on the robustness of the model, the batch size varies from 32,64,128,216 to 512, but from the performance effect, their improvement or reduction of the model is not obvious. [3]

4) Xavier Initialization: At the beginning of the experiment, we do not use the method of the Xavier initialization. The performance of the classification model is shown in figure 11.

Figure 11 shows the relationship between cross entropy loss value and the number of iterations. As the iterations increase, the loss value of our classification model has no obvious trend of convergence.

As shown in figure 12 & 13, the performance of train and validation data-set and the loss value is approximately above 0.7, even above 0.8 in some cases, and the accuracy is approximately stable between 0.48-0.50. It can be seen that without the Xavier initialization, our classification performance has dropped significantly. So using the Xavier initialization can significantly improve the classification performance of our

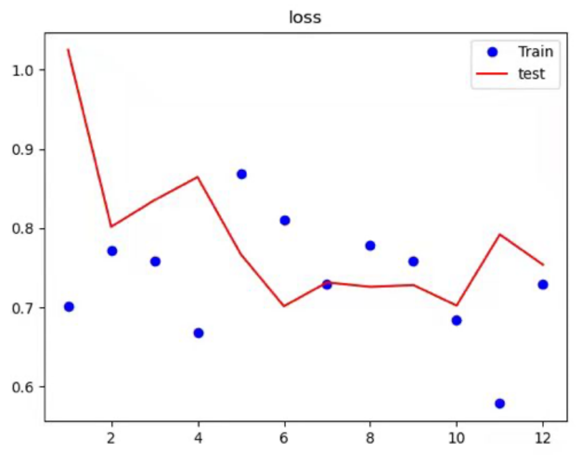


Fig. 10. Learning Rate = 1×10^{-5}

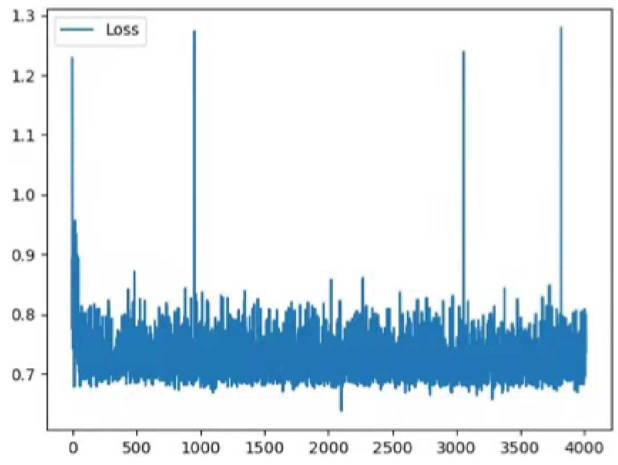


Fig. 11. Cross Entropy Loss Value VS Number of Iterations(without Xavier)

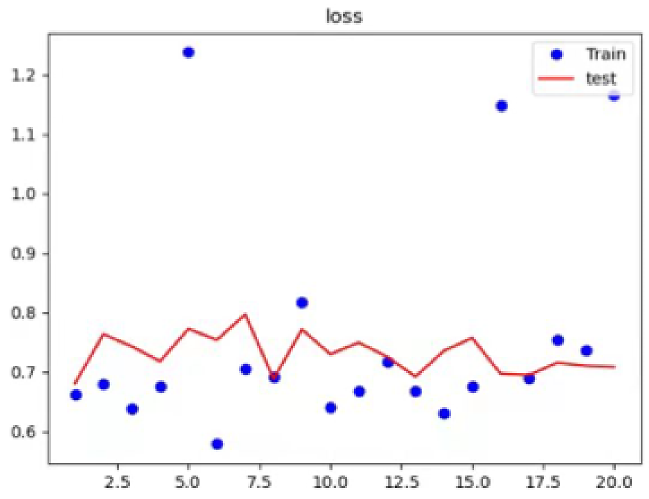


Fig. 12. Cross Entropy Loss VS Number of Epochs(without Xavier)

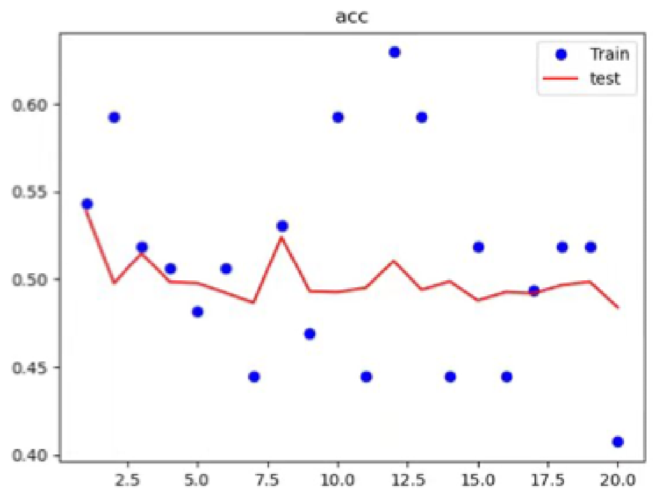


Fig. 13. Accuracy Changes VS Number of Epochs(without Xavier)

model.

B. Return Prediction

We use the same 20-day horizon images to train our model to predict the return trend of subsequent 20-days, due to Network and GPU limitations, we did not make any other prediction for 5-days or 60-days. The performance of the prediction is shown in figure 14 & 15.

As we can see, the number of epoch and iteration is even larger than when we do classification. Although CNN has improved data requirements compared to traditional neural networks, it still has a large demand for data. At the same time, the existence of the pooling layer will cause the loss of a lot of very valuable information, while also ignoring the relationship between the whole and the part. which may also cause the convergence of prediction process to be relatively slow.

For the prediction robustness, we analysis the impact of changing the dropout rate on the robustness of the model by changing dropout rate from 0 to 50%, the performance of the prediction model is as follows:

When we keep learning rate at 1×10^{-3}, batch size at 128, without dropout and the Xavier initialization, the performance is shown in figure 16 & 17. [4]

When we keep learning rate at 1×10^{-3}, batch size at 128, dropout rate at 50% and without the Xavier initialization, there is still no obvious performance improvement of our prediction model.

VI. Conclusion

Based on the comparison of the final results from multiple sets of experimental parameters, we found out that different

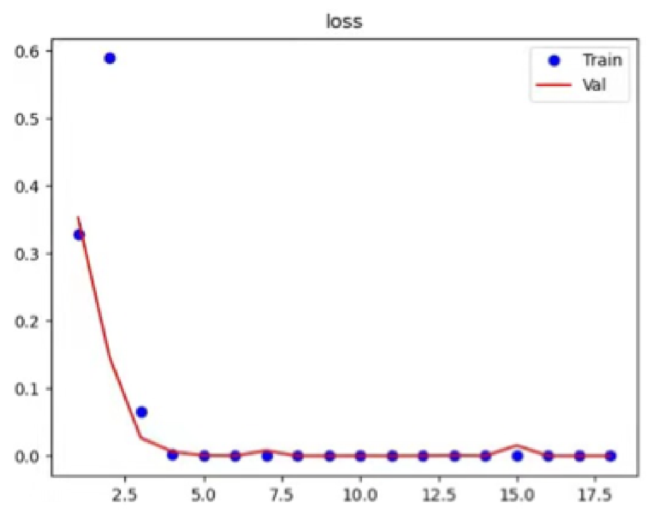


Fig. 14. Cross Entropy Loss VS Number of Epochs(Prediction)

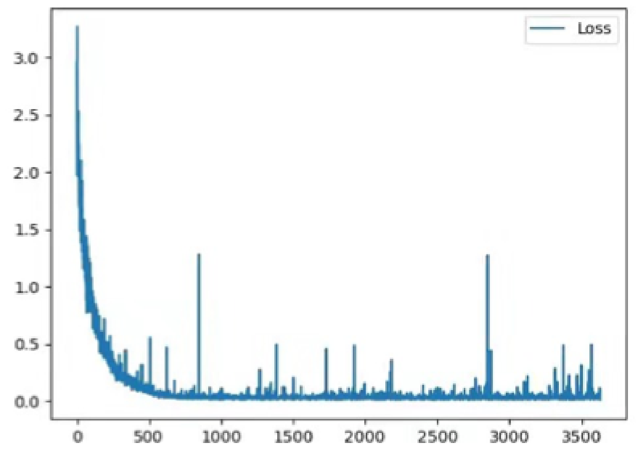


Fig. 15. Cross Entropy Loss Value VS Number of Iterations(Prediction)

parameters only have little effect on the experimental results. It may be caused by the model itself, since the model is not complicated enough. Along with the potential reason that there are plenty of picture features. Which all make it difficult to accurately fit the model. From the data perspective, the image signal-to-noise ratio is too high and the convolutional network failed to specifically extract useful information. instead, a unified convolution operation, which causes noise to be added and further affects the final accuracy. Furthermore, from the label and input perspective, the correlation between the input data and the label is not that strong, while the timing signal is not stable as well, which are also problems that cause the model to be less accurate. In the end, our team successfully reproduce this report and achieve a classification result of about 50% by strictly follow the parameters process mentioned in the report.

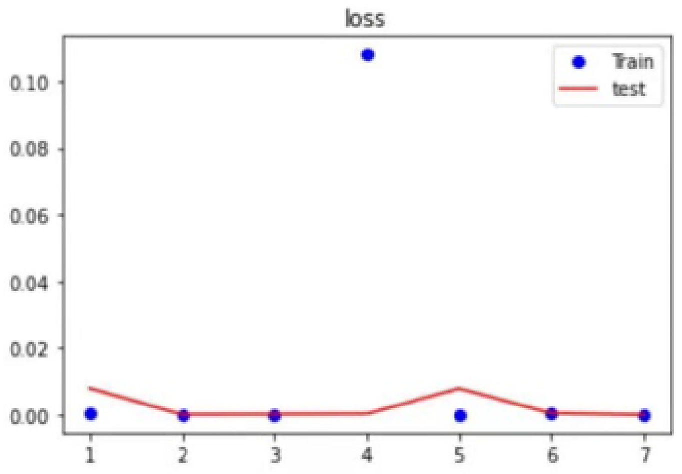


Fig. 16. Cross Entropy Loss VS Number of Epochs(Prediction without Dropout and Xavier)

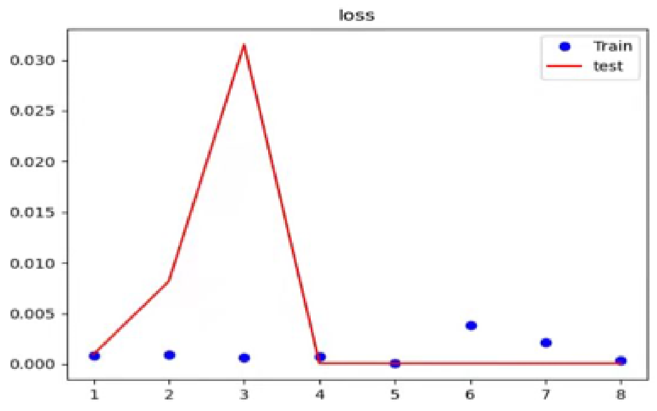


Fig. 17. Cross Entropy Loss Value VS Number of Iterations(Prediction without Dropout and Xavier)

VII. Team Work Illustration

A. *Members*

- Xinzhen Liao (20813829)
- Yuan Xu (20801498)
- Rui Ma (20736954)
- Yiyuan Shi (20745230)

B. *Contribution*

The whole team actively discuss the entire structure and idea for this project. Xinzhen Liao and Yuan Xu mainly in charge of the coding part, while Rui Ma and Yiyuan Shi are responsible for report writing. Although each of our teammate has different obligations, all of us help each other and study together through the whole project.

References

[1] Jiang, Jingwen and Kelly, Bryan T and Xiu, Dacheng, "(Re-) Imag (in) ing Price Trends," 2020.

[2] “Everything You Should Know About Dropouts And BatchNormalization In CNN,” Sep. 14, 2020. [Online]. Available: https://analyticsindiamag.com/everything-you-should-know-about-dropouts-and-batchnormalization-in-cnn/. [Accessed: Sep. 14, 2020].

[3] Johann Huber, “Batch normalization in 3 levels of understanding,” Nov. 07, 2020. [Online]. Available: https://towardsdatascience.com/batch-normalization-in-3-levels-of-understanding-14c2da90a338. [Accessed: Nov. 07, 2020].

[4] Jason Brownlee, “Dropout Regularization in Deep Learning Models With Keras,” Jun. 20, 2016. [Online]. Available: https://machinelearningmastery.com/dropout-regularization-deep-learning-models-keras/. [Accessed: Jun. 20, 2016].