Cryptocurrency Price Distribution Prediction Using Neural Networks

Robert Ciborowski
University of Toronto
robert.ciborowski@mail.utoronto.ca

Yunfei Ouyang
University of Toronto
yunfei.ouyang@mail.utoronto.ca

Abstract

Price prediction in the financial market is attractive for both holders and traders due to its volatility and uncertainty. Many have tried to analyze the future of the financial market by using methods and models from machine learning. In this paper, we construct several neural network models to predict sequential data for cryptocurrencies, specifically Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), CNN-RNN Multilayer Perceptron (CNN-RNN-MLP), Long Short-Term Memory with Attention (LSTM-Attention), and InceptionNet. The results showed that CNN-RNN-MLP performs the best among other models.

1 Introduction

Financial time series prediction is a major topic within the finance industry. It is a difficult task due to the uncertainty of the financial market. In recent years, the strategies of predicting financial assets have expanded. Researchers have proposed many state-of-the-art methods for stock and cryptocurrency predictions, such as sequential prediction using the CNN and LSTM [4]. However, current research has not proposed models which give traders buying and selling points. Current research also does not explore newer machine learning model types and does not take advantage of technical indicators used by professional human traders. In this paper, we are going to focus on the forecast of cryptocurrencies using several neural network techniques. First, we train our models with Bitcoin data, where the output of our models is a percentile of our choosing of the distribution of Bitcoin prices of the next 12 hours, thus providing information for traders on when to buy and sell Bitcoin in the short term. Then we use the top performance models to perform extensive experiments on different cryptocurrencies and on different percentiles. Lastly, we compare our model architectures against those from other research.

2 Related Work

The recent trend of predicting financial assets is to use machine learning techniques, where deep learning neural networks are heavily utilized. As outlined in [1], there are two main research approaches to improve the price movement prediction. One is feature extraction. They proposed that feature extraction with ANN, SVM and CNN can result in better prediction performance. Currently, CNN is the main architecture used for feature extraction in the finance industry. As in [2], the authors used candlestick charts instead of price data as the inputs to the CNN for feature extractions. In [3], the paper proposed to use a 3-dimensional tensor to represent the input data, then feed it to the CNN. Another research approach mentioned in [1] is model enhancement, where combining multiple models may yield better prediction accuracy. [4] finds that the CNN-LSTM model outperforms both CNN and LSTM in the prediction task, and [5] proposed a Multi-filter Neural Network, which integrates both convolutional and recurrent neurons to construct the multilayer filter structure. Lastly, [6] used CNN, LSTM, and Bi-LSTM to predict the price of cryptocurrencies, where they found some limitations around those advanced deep algorithms.

Preprint. Under review.

3 Data

Our algorithm takes 15 days' worth of time-series data of cryptocurrencies as inputs. The data points are partitioned into 0:00am-2:59am, 3:00am-5:59am, ..., 9:00pm-11:59pm sections of trading hours. Our inputs contained 120 data points for 15 days. A single data point contains information about the asset's price, volume, and 7 technical indicators outlined in [A1]-[A6], which allow our models to incorporate short and long term price momentum, volume-weighted price momentum and underbought/overbought signals. Unlike other research, which predicts if the price will be higher or lower, our models predict one of 15th, 25th, 35th, 50th, 65th, 75th or 85th percentiles of the 12-hour price to provide good buying and selling points of Bitcoin for traders.

We normalized our data so that every input and output was between 0 and 1. Our outputs were normalized by dividing by the last input day's mean price, and then dividing by 2. This means that if the predicted percentile was 10% greater than the last input day's mean price, our value would be 0.505 because 0.505 $\times$ 2 = 1.1, and the predicted percentile is equal to $1.1 \times average\_price_{last\_day}$. Normalizing our outputs between 0 and 1 allows us to experiment with a larger variety of activation functions for the output layer.

4 Models

4.1 CNN Model

The CNN model consists of four blocks. The first block contains a 1D convolution layer, a 1D max-pooling layer, a batch normalization layer, and a sigmoid activation layer. The second block contains a 1D convolution layer, and a 1D max-pooling layer. The third block contains a 1D convolution layer, a 1D max-pooling layer, and a batch normalization layer. The final block contains a flatten layer, a dense layer, a batch normalization layer, and a dense layer. This model is inspired by the CNN in computer vision, as well as the outline in [1].

4.2 RNN Model

The RNN model consists of a bi-directional LSTM layer, as well as a dense layer. We decided to use a bi-directional LSTM instead of a forward LSTM because the bi-directional property learns from the future and the past input data. We want to see how the LSTM performs without any other layers, and we expect the bi-directional LSTM to catch patterns in the data better than the CNN architecture above.

4.3 CNN-RNN-MLP Model

The CNN-RNN-MLP model consists of a 1D convolution layer, a bi-directional LSTM layer, and multiple layers of perceptrons. The motivation behind this combination of layers was for the convolutions to learn patterns in the inputs, for the LSTM units to measure sequences and distances between patterns, and for the perceptrons to take information from the LSTMs to make a final judgement on what the predicted output should be. Adding multiple convolution layers did not help our model fit the data any better than a single convolution layer with a kernel size of 9, so we decided to use just one convolution layer.

4.4 LSTM-Attention Model

The LSTM-Attention model is an enhancement of the RNN model and consists of a bi-directional LSTM layer, a sequential multiplicative attention layer, and a flatten layer paired with a dense layer. The model is inspired by the attention mechanism in natural language computing, where the model learns to attend to a sequence of words. In our case, the model learns to attend to a sequence of prices. However, instead of using additive attention, we used multiplicative attention as implemented by the Keras Self-Attention library [A7], which can be expressed as follows:

$$e_{t,t'} = \sigma(x_t^T W_a x_{t'} + b_a) \tag{1}$$

4.5 InceptionNet Model

The InceptionNet model consists of a single inception layer, a max-pooling layer and multiple layers of perceptrons. The inception layer was inspired by Google's inception module with dimensionality reduction [7], which contains convolution layers and a max-pooling layer placed side-by-side so that their outputs are concatenated together. This layer also takes advantage of dimensionality reduction, which involves using convolutions with 1x1 kernels before our other convolutions. Dimensionality reduction allows our model to utilize extra computationally inexpensive convolutions.

5 Experiments/Results/Discussion

There are two sets of experiments we are going to conduct. The first experiment is to test our models on the prediction performance of the 15th percentiles of the 12-hour price distributions on Bitcoin. The second experiment will use models with good stability and prediction performance from the first experiment. We will do extensive experiments on those models with different percentiles and different cryptocurrencies. The performance metric for our models is Mean Absolute Error (MAE), which is expressed as follows:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - t_i| \tag{2}$$

Where N is the total sample size, y_i is the ith prediction and t_i is the ith label. See [A8] for details on the meaning of MAE and our models' outputs. Although our data is based on the volatile prices of cryptocurrencies, a greater than 2% difference between the predicted and actual values is high. A difference of less than 2%, or an absolute error of less than 0.01, is preferred.

5.1 Initial experiment

Here are the results for our models on the prediction performance on Bitcoin:

Prediction performance on Bitcoin						
Model	Train Loss	Train MAE	Val Loss	Val MAE	Test Loss	Test MAE
CNN	8.0203e-05	0.0067	1.1598e-04	0.0087	3.5781e-04	0.0148
RNN	2.3168e-04	0.0099	1.0877e-04	0.0073	3.7976e-04	0.0144
CNN-RNN-MLP	2.8405e-05	0.0040	1.5543e-04	0.0093	3.0706e-04	0.0138
LSTM-Attention	2.0168e-04	0.0100	9.2066e-05	0.0068	3.3111e-04	0.0137
InceptionNet	1.9362e-06	0.0011	1.2158e-04	0.0086	3.5276e-04	0.0147

During the experimentation, we found that using only RSI-1 (14 day period), EMA and MFI obtains similar loss as above, where we used all the indicators. We discovered that the LSTM-Attention's validation loss curve model is very unstable,[A12] possibly because Bitcoin is very volatile. Moreover, we found that CNN, RNN and LSTM-Attention converge quickly,[A9][A10][A12] where CNN-RNN-MLP and InceptionNet have a much lower MAE and are relatively stable when compared to other models.[A11][A13] Thus we will use these two models for extensive experiments. After this experiment, an enhancement we applied on CNN-RNN-MLP and InceptionNet is weight regularization, where we used L2 regularization for both models to overcome their overfitting issues [A14][A15].

5.2 Extensive experiments on the best models

After the first experiment, we decided to do extensive testing on the L2-regularized CNN-RNN-MLP and InceptionNet models. We will train these models with 15th, 50th, and 85th percentiles on Bitcoin to see if the models perform better or worse when predicting percentiles far away from the mean. Here are the results:

Best models prediction performance on different percentiles on Bitcoin							
Percentile	Model	Train Loss	Train MAE	Val Loss	Val MAE	Test Loss	Test MAE
15th	CNN-RNN-MLP	4.1248e-04	0.0067	3.6804e-04	0.0060	4.8671e-04	0.0103
15th	InceptionNet	4.7643e-04	0.0128	2.8633e-04	0.0104	3.7659e-04	0.0124
50th	CNN-RNN-MLP	2.8220e-04	0.0065	2.4655e-04	0.0059	3.8206e-04	0.0104
50th	InceptionNet	4.7622e-04	0.0129	3.1858e-04	0.0114	4.0264e-04	0.0131
85th	CNN-RNN-MLP	3.0583e-04	0.0068	2.8740e-04	0.0068	4.6348e-04	0.0119
85th	InceptionNet	4.8825e-04	0.0136	3.1350e-04	0.0119	4.1831e-04	0.0140

We see that both models can predict several percentiles of Bitcoin prices equally well, where the CNN-RNN-MLP model has an overall lower test MAE. We believe that this is due to the LSTM units of the CNN-RNN-MLP model, since InceptionNet only has convolution layers and max-pooling layers. We discovered that LSTM can help the model to analyze the price pattern in the data, which helps to stabilize the model prediction as well as the testing MAE.

We also noticed that our models obtain similar losses on the training and validation datasets with regularization, where the validation dataset consists of data from the two months after the training data. However, the models achieve greater loss on the test datasets. We hypothesize that this is occurring because the test dataset consists of Bitcoin data from January to March of 2021, which was a period in which Bitcoin was making news headlines and gaining investments from large institutions. Thus, the market behaviour of Bitcoin may have changed significantly in 2021, causing our model, which was trained on 2018-2020 data, to perform poorly on the test dataset.

Moreover, we trained the CNN-RNN-MLP model and the InceptionNet model on the 50th percentiles of other major cryptocurrencies: Ethereum (ETH) and Litecoin (LTC). Here are the results:

Best models prediction performance on 50th percentiles on different cryptocurrencies							
Cryptocurrency	Model	Train Loss	Train MAE	Val Loss	Val MAE	Test Loss	Test MAE
ETH	CNN-RNN-MLP	6.8847e-04	0.0113	6.3349e-04	0.0111	9.4356e-04	0.0165
ETH	InceptionNet	4.7643e-04	0.0128	2.8632e-04	0.0104	7.2074e-04	0.0181
LTC	CNN-RNN-MLP	5.7860e-04	0.0107	8.8684e-04	0.0172	8.3224e-04	0.0167
LTC	InceptionNet	4.0983e-04	0.0125	5.1633e-04	0.0142	6.9593e-04	0.0188

We see that our models work with different major cryptocurrencies, which is expected since movements in Bitcoin, the cryptocurrency with the largest market cap, tend to translate to movements in other cryptocurrencies as well.

5.3 Comparisons

Our models were designed to predict percentiles. However, other cryptocurrency prediction papers focused on predicting whether the next 24-hour price will be higher or lower, which is a binary prediction. In order to compare our model architectures against architectures of other models, we modified our models to make these binary predictions. We also changed our train and test datasets to not include data from 2021 as other papers did not have access to this data. We are comparing against the best performing model from [6], a CNN and bi-LSTM model, which is similar to our CNN-RNN-MLP but uses more convolution layers and does not include any technical indicators as inputs. We are also comparing against the highest accuracy model from [8], a deep neural network. Here are our results:

Performance comparisons between other papers' models and our models			
Model	Test Accuracy	Test Percision	Test Recall
CNN-RNN-MLP	0.7107	0.7265	0.9038
Inceptionnet	0.6670	0.6881	0.5232
[6]	0.5543	unknown	unknown
[8]	0.5306	0.5290	0.6970

6 Conclusion

This paper uses 3-hour cryptocurrency data to predict the behaviour of a cryptocurrency over the next 12 hours. Unlike other research, this paper takes advantage of technical indicators used by professional traders to improve prediction performance. Instead of predicting whether the price will be higher or lower, our models are designed to predict percentiles of the next 12-hour price distribution, providing traders with more useful information: buying and selling points.

We found that our CNN-RNN-MLP and InceptionNet model architectures perform the best on new data. Additionally, these architectures outperform the state of the art cryptocurrency architectures. However, due to the ongoing changes in the behaviour of cryptocurrency markets, the data which our models are trained on differ from current market trends, thus limiting the performance of our models on recent data. Our models, which focus on technical analysis, could be improved by taking into account events in the news, such as Tesla Inc.'s Bitcoin investment or the Security and Exchange Commission's upcoming cryptocurrency money-laundering regulations. Our future work would be to include market sentiment analysis to improve the performance of our models.

Contributions

Robert contributed to the data collection, the CNN-RNN-MLP model and the InceptionNet model.
Yunfei contributed to the CNN model, the RNN model, and the LSTM-Attention model.
We contributed equally to the report.
The git repository to this project can be found at
https://github.com/Robert-Ciborowski/CSC413-Final-Project

References

[1] Can Yang, Junjie Zhai , and Guihua Tao, "Deep Learning for Price Movement Prediction Using Convolutional Neural Network and Long Short-Term Memory," Mathematical Problems in Engineering 2020: pp. 1-13, 16 July 2020.

[2] S.-J. Guo, F.-C. Hsu, and C.-C. Hung, "Deep candlestick predictor: a framework toward forecasting the price movement from candlestick charts," in Proceedings of the 2018 9th International Symposium on Parallel Architectures, Algorithms and Programming (PAAP), pp. 219–226, IEEE, Berlin, Germany, 2018.

[3] E. Hoseinzade and S. Haratizadeh, "Cnnpred: cnn-based stock market prediction using a diverse set of variables," Expert Systems with Applications, vol. 129, pp. 273–285, 2019.

[4] C. Li, X. Zhang, M. Qaosar, S. Ahmed, K. M. R. Alam, and Y. Morimoto, "Multi-factor based stock price prediction using hybrid neural networks with attention mechanism," pp. 961–966, IEEE, Berlin, Germany, 2019.

[5] W. Long, Z. Lu, and L. Cui, "Deep learning-based feature engineering for stock price movement prediction," Knowledge-Based Systems, vol. 164, pp. 163–173, 2019.

[6] Pintelas E, Livieris IE, Stavroyiannis S, Kotsilieris T, Pintelas P. Investigating the Problem of Cryptocurrency Price Prediction: A Deep Learning Approach. Artificial Intelligence Applications and Innovations. 2020;584:99-110. Published 2020 May 6. doi:10.1007/978-3-030-49186-4_9

[7] C. Szegedy et al., "Going deeper with convolutions," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 2015, pp. 1-9, doi: 10.1109/CVPR.2015.7298594.

[8] Ji, Suhwan, Jongmin Kim, and Hyeonseung Im. "A Comparative Study of Bitcoin Price Prediction Using Deep Learning." Mathematics 7, no. 10 (2019): 898. https://doi.org/10.3390/math7100898.

Appendix

[A1] This table shows information on the technical indicators that we used as additional inputs to our models.

Technical Indicators Used As Model Inputs			
Name	Short Name	Parameters	Type
Relative Strength Index	RSI-1	14-day	Momentum, Over/Underbought Signals
Relative Strength Index	RSI-2	42-hour	Momentum, Over/Underbought Signals
Exponential Moving Average	EMA	63-hour	Momentum
Moving Average Convergence Divergence	MACD-1	12/26-day	Momentum
Moving Average Convergence Divergence	MACD-2	3/6.5-day	Momentum
Money Flow Index	MFI	42-hour	Volume-weighted momentum
Bollinger Bands®	BOLL	20-day	Over/Underbought Signals

[A2] https://www.investopedia.com/terms/r/rsi.asp

[A3] https://www.investopedia.com/terms/e/ema.asp

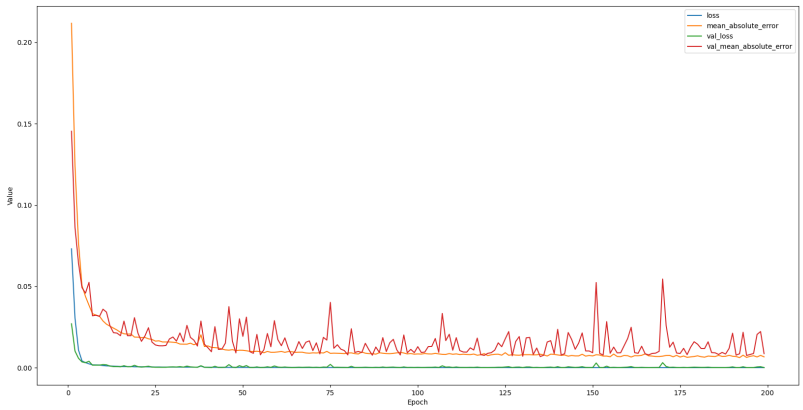
[A4] https://www.investopedia.com/terms/m/macd.asp

[A5] https://www.investopedia.com/terms/m/mfi.asp

[A6] https://www.investopedia.com/trading/using-bollinger-bands-to-gauge-trends/

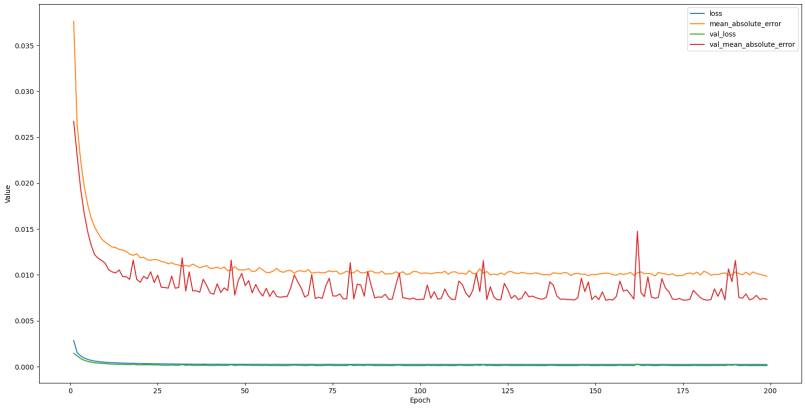
[A7] https://pypi.org/project/keras-self-attention/

[A8] The mean absolute error metric shows the mean absolute difference between the model predictions and model results. If a model predicts a value of 0.52, and the actual value was 0.51, then the absolute error would be 0.01. In this case, the model predicted that the 15th percentile was equal to $(0.52 \times 2) \times average\_price_{last\_day} = 1.04 \times average\_price_{last\_day}$, or that the 15th percentile was 4% greater than the 15th day's average price. The actual percentage increase was only 2% greater than the 15th day's average price, so a mean absolute value of 0.01 means that the model's predicted increase was different by 2%.



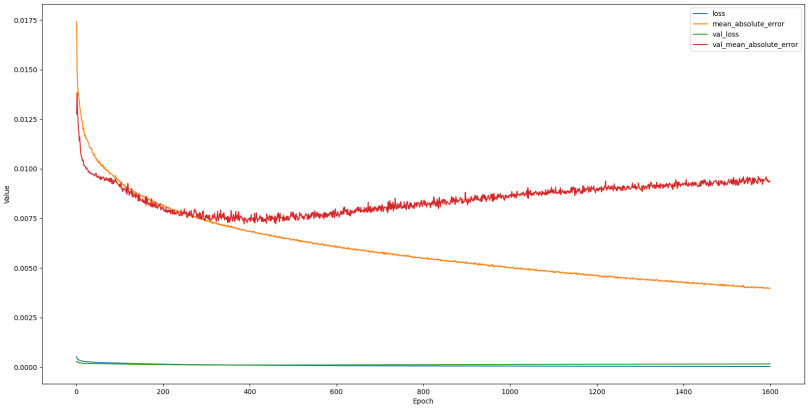
[A9]

Figure 1: Initial experiment CNN model loss plot



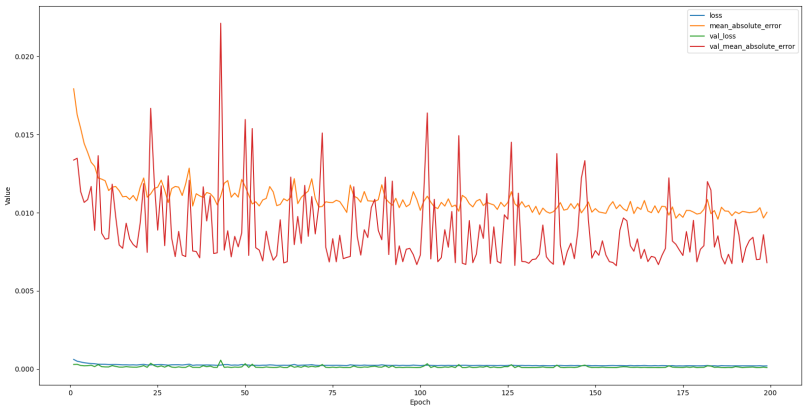
[A10]

Figure 2: Initial experiment RNN model loss plot



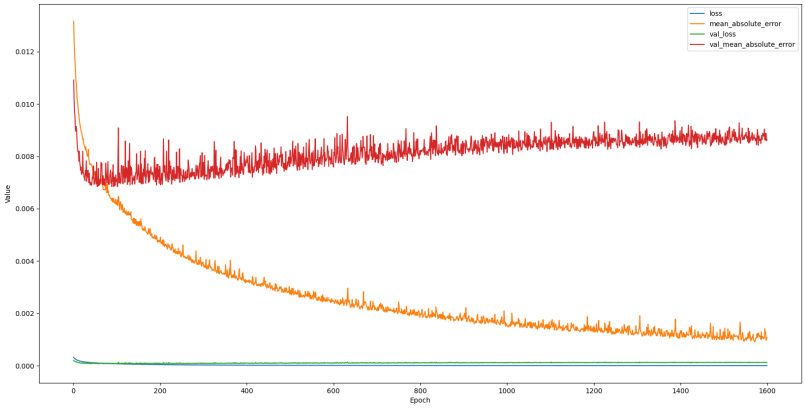
[A11]

Figure 3: Initial experiment CNN-RNN-MLP model loss plot



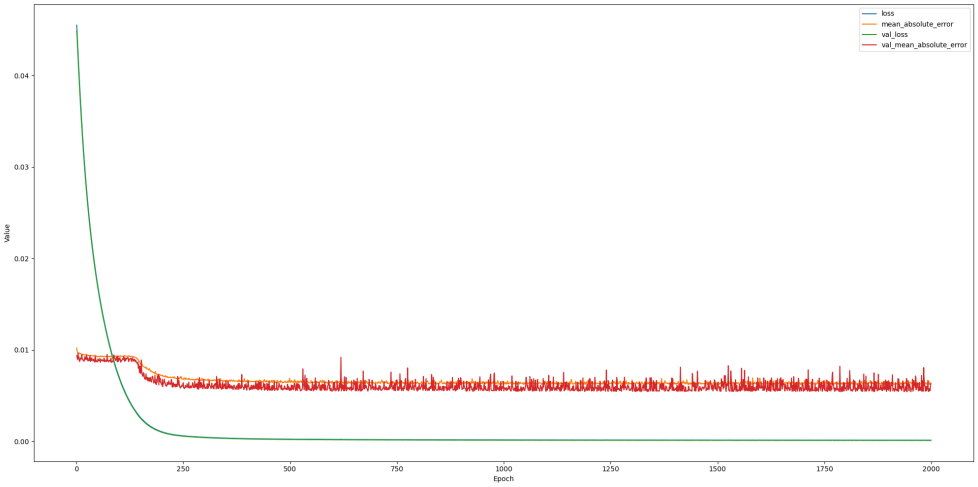
[A12]

Figure 4: Initial experiment LSTM-Attention model loss plot



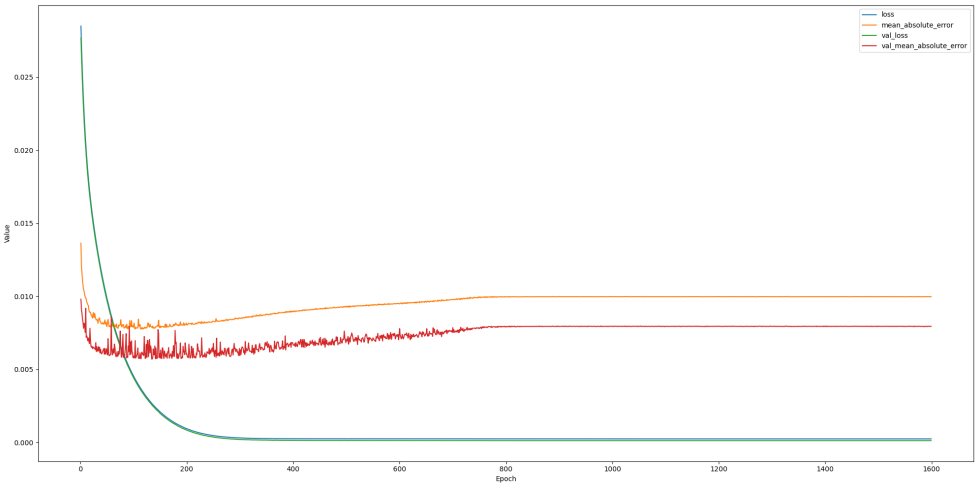
[A13]

Figure 5: Initial experiment InceptionNet model loss plot



[A14]

Figure 6: regularized CNN-RNN-MLP model loss plot



[A15]

Figure 7: regularized InceptionNet model loss plot