Predicting the Demand in Bitcoin Using Data Charts: A Convolutional Neural Networks Prediction Model

Ahmed.F. Ibrahim
Computer Science Department
University of Waterloo
Waterloo, Ontario, Canada
a24ibah@uwaterloo.ca

Liam Corrigan
Management Science Department
IVEY Business School
London, Ontario, Canada
lcorrigan.msc2018@ivey.ca

Rasha Kashef
Electrical, Computer, Biomedical
Egineering Department
Ryerson University
Toronto, Ontario, Canada
rkashef@ryerson.ca

Abstract—**Traditional time series modeling techniques emphasize on predicting cryptocurrencies using classically structured data representation as numerical features to present the time-series datasets. In this paper, a novel approach to analyze time-series data charts using a modified Convolutional Neural Networks (CNNs) is proposed. The CNNs have been adopted to recognize subtle and undetectable patterns within images of time-series data charts. Our approach has been proven to achieve significant results, suggesting a need for further research into this new method for time series modeling, especially for Bitcoin.**

Keywords—Bitcoin, Price Prediction, Time-series, Data charts, Deep learning, CNN.

I. Introduction

In the world of stock trading, traders are looking for patterns like accessing triangles, head and shoulders, double tops, and Elliot waves. E-traders claim that these patterns can be used to predict the future market movement and guide their trading strategy [1]. Deep Learning has demonstrated a significant ability to recognize subtle, undetectable patterns in various applications, including stock market prediction [2]-[8]. With the impressive ability of Convolutional Neural Networks (CNNs) in detecting subtle, difficult to find, patterns in images, it is believable that the CNNs can detect those hidden patterns within images [9]-[13], especially candlestick charts [14] and use these patterns to predict future market movement. Wang and Oates [8] have applied CNNs in predicting product demand by encoding the time series into Gramian Angular Fields and Markov Transition Fields. Their model has shown very competitive results when compared to five state-of-the-art models. The idea of forecasting demand from image representations of data was motivated by [14][15], which involves the same up/down classification problem for Bitcoin. In [15], the Tensorflow and the AlexNet architecture [11], [13] for the CNNs model is used. Experimental results in this work claimed to achieve over 70% accuracy on the up/down binary classification for Bitcoin prices. The work in [15] did not test the model on unseen data. It uses the validation set for accuracy, which creates cause for concern. In [10], the data creation process creates test and validation sets from random windows within the same time period, meaning the model can see possible validation images in the training set. This would give the model foresight during training, making the validation accuracy less meaningful. Inspired by [8], [10], [14], and [15], this paper aims at improving the prediction accuracy of cryptocurrencies' prices with a focus on Bitcoin using time-series data charts. In this paper , we are using a more advanced architecture, ResNet, and implementing stochastic gradient descent with restarts, and cyclical learning rate selection [10], [12]. This paper separates the test data from the training and validation data to better assess the accuracy. The proposed model has achieved an accuracy of 78.6%, which shows a significant improvement in analyzing time-series data charts instead of traditional feature-based time-series data.

The rest of the paper is organized as follows: In section 2, related work on cryptocurrency prediction models is presented. Section 3 introduces the CNNs. Section 4 discusses the proposed model using the modified architecture of the CNNs. Experimental results and validations are explained in Section 5. Finally, the conclusion and future directions are discussed in Section 6.

II. Related Work and Background

Cryptocurrencies, such as Bitcoin, Ethereum, and Litecoin are an alternative class of digital assets that are primarily used as a medium of exchange [12],[14],[17],[16]. Cryptocurrencies and Stock price predictions have been a heavily studied topic for decades. Traditional time series modeling has generally shown marginally positive results, at best. It can usually be concluded that the randomness of stock prices cannot be predicted using traditional machine learning techniques. While more traditional methods like ARIMA [17][18][19][20] appear not to work when it comes to stocks, new techniques and neural network architectures might prove to have greater predictive power than previous modeling techniques. Long Short-Term Memory (LSTM) [16], recurrent neural networks (RNN) [21] are one cutting edge architecture that has been showing a significant progress in the field of time-series predictions. With the incredible accuracy being achieved using deep learning, in particular, using CNNs, new research directions [2]-[8] have been investigated using deep neural networks that achieve incredible feats and breakthroughs especially in complex image recognition applications [9]-[13].

III. The Proposed CNN Model using RESNET34

The CNNs operate by reacting to input, passing that reaction forward to further neurons, and training a receptive field to interpret the response and begin to make predictions. The CNNs are typically implemented in a series of alternating layers. These alternating layers are generally ordered in such a way that they have convolutional layers alternating with pooling layers. Pooling layers reduce the number of free variables at the end of the process that gets passed on to the receptive field that is the trainable part of the network. If a network applies pooling layers that shrink the depth of the problem in between these convolutional layers, it can be considered a local pooling layer. If the pooling happens at the end of the convolutions, it is called global. The more convolutional layers to the network, the "deeper" it is considered. CNNs are used to process visual data with the ability to interpret spatially linked data. Analyzing time-series datasets using the data chart is a recent effort in the last few years. In this paper, we show the importance of using a visual representation of data in providing better prediction of those hidden patterns in bitcoin trends using deep learning with a focus on CNN. The proposed model uses a CNN architecture known as resnet34 [11] and PyTorch [22], a dynamic numerical computation framework made by Facebook as its competitor to Google Tensor Flow. The process for model development involves pre-trained neuron weights calculations based on the winning ImageNet submission, determining of an optimal learning rate, training the last few layers of the network to get a base set of weights, and training the entire network until overfitting started to occur. The learning method optimized the log loss using stochastic gradient descent (SGD) with restarts [10] and a cosine annealing function.

A. Precomputed Weights

Not using data augmentation only gave the extra benefits of precomputing and saving layer outputs for each image, which helps improve future training steps. Each image in the test and validation sets were run through the network using the default set of weights associate with resnet34. Outputs coming out of the second last layer were saved as the precomputed inputs to the final layer.

B. Choose Learning Rate

Choosing the learning rate should be low enough to ensure convergence. However, if it is too low, there is a risk that the gradient optimization might get stuck in a local minimum. Finding the learning rate works by iteratively decreasing the learning rate until performance starts to degrade. The learning rate was then selected to be a magnitude of 10 larger than the optimal learning rate. In this case, the optimal learning rate was 10^{-5}, so the learning rate was set to 10^{-4}. This choice has been made to benefit the SGD [10] with restarts algorithm and help reduce overfitting during earlier training cycles. Cosine annealing decreases the learning rate iteratively between mini-batches. Between cycles, the learning rate is reset, such that the model escape from narrow valleys in the multi-parameter optimization space as shown in Fig.1. Wider yet shallower valleys lead to better generalization.

C. Partially Trained network

The network was initialized with the weights from the resnet34 architecture, trained on the ImageNet dataset. The training was focused on only the final layer for the first four cycles to leverage the pre-learned features present in the beginning layers. This helped the network train faster overall. This benefited from the pre-computed set of weights where the saved outputs from the second last layer were fed into the final layer for training, avoiding the need to rerun the entire network on each image.

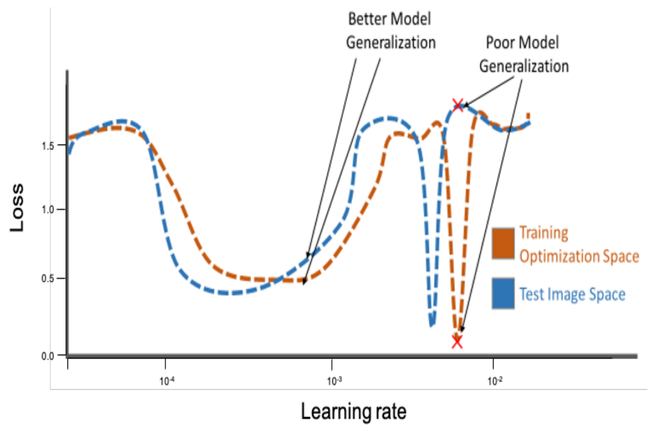

Fig. 1. Wide Valleys Lead to Better Model Generalization

D. Fully Trained Network

Once the final layer had been trained, the previous layers in the network were unlocked, and the full network was trained using increasing learning rates. The first third of the network layers used $\frac{10^{-4}}{9}$ as the learning rate. The following two-thirds used $\frac{10^{-4}}{3}$ and 10^{-4}, respectively. Lower learning rate helps prevent the SGD algorithm from moving too far from its pre-trained weights, so the effect of "stepping out of valleys" is not as powerful and the ability of the early layers to capture generalized image features remains. The final training used a cycle multiplier so that the restarts in the SGD with restarts algorithm would occur less frequently. The learning rate is decreased following the cosine function for three cycles consisting of 1, 2, and 3 full epochs, for a total of 6 epochs of training as shown in Fig.2.

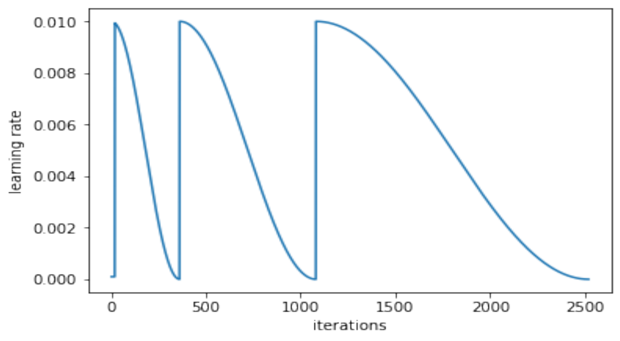

Fig. 2. Learning Rate for Full Training

IV. Experimental Analysis and Results

A. Datasets

The input images for the convolutional neural network are split into a training/validation and testing parts. Table 1 illustrates the periods selected for these parts.

TABLE I. Periods selection for training/testing datasets

Split	Date Range	5-minute Periods
Training and validation	January 27, 2015, to February 06, 2018	318,528
Testing	February 06, 2018, to March 23, 2018	14,400

The images used for training the CNN were generated by randomly selecting a number between 1 and $318{,}528 - 41$, then taking the following 40 periods. The number of periods was selected as a simple random sampling. Further, an open-high-low-close chart (also OHLC) [23] was created for these 40 periods. An OHLC chart is a bar chart that indications the open, high, low, and closing prices for each period. Next, the generated images were cropped to remove extra white space, and get rid of the prices along the y-axis. Square images are essential for improving the speed of matrix multiplication in the GPU. This is due to an issue with the CUDA framework for Nvidia GPUs and exists for Tensor Flow 1.7. Examples of these images are illustrated in Fig.3.

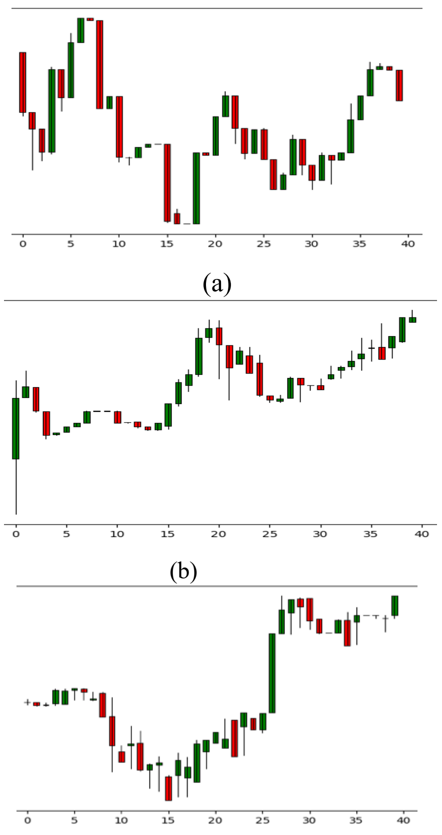

Fig. 3. Three Candlestick Price Charts Spanning 40 5-Minute Periods

Each image was then classified as either UP or DOWN by comparing the close prices for the $40^{th}$and 41^{st} periods. The process was repeated 500,000 times. Every 10th image created was placed into a validation set used to measure the log-loss during training. Giving 450,000 images for training and 50,000 for validation. The test images we generate using sensed data. These images were created using a sliding window covering every possible period between February 06, 2018, and March 23, 2018. This resulted in $14{,}400 - 40 - 1 = 14{,}359$ images for the initial test set (Coinbase data). To further test the model, 50,000 images were generated from the data taken from Poloniex. The period for many of these images overlaps with the data that was used for training (Sept 2017 – December 2017). A set of 3908 images was generated for Apple, Facebook, Google, and Microsoft stocks spanning from January 1st, 2018, to March 23rd, 2018 to checks if training the model can make predictions on stock data.

B. Training the CNN

Images were fed into the network at a resolution of 480x480 to ensure network stability with higher accuracy. For most steps, the batch size was 64. However, this had to be lowered to 16 during the final stage due to memory limitations within the GPU. Training originally took 13 days in total on an intel i7 7700k server with Nvidia 1050Ti GPU (about a 3x performance boost over and Amazon Web Services P2. xlarge instance).

C. Accuracy of the Proposed CNN Model

The proposed CNN network has shown accuracies of 75.74%, 74.74%, 77.69%, 77.94%, 77.66%, 77.56% for Coinbase, Poloniex, Apple Stock, Facebook, Google Stock, Microsoft, respectively. However, the images in [15] had been classified incorrectly by looking at the second last close price rather than most recent as illustrated in Eq1. and Eq2.

$$\text{class}_{t=41} = \begin{cases} \text{UP}, & \text{if } \text{price}_{t=41} - \text{price}_{t=39} \\ \text{DOWN}, & \text{otherwise} \end{cases} \quad (1)$$

$$\text{class}_{t=41} = \begin{cases} \text{UP}, & \text{if } \text{price}_{t=41} - \text{price}_{t=40} \\ \text{DOWN}, & \text{otherwise} \end{cases} \quad (2)$$

A pull request was submitted to the original repository that solves the data generation problem (Fig.4). To retrain the model using the correct image classes, the same learning rate selection procedure and pre-training were followed. After training on the correct classes, the model achieved the highest accuracy with, 78.60% on the Coinbase test set.

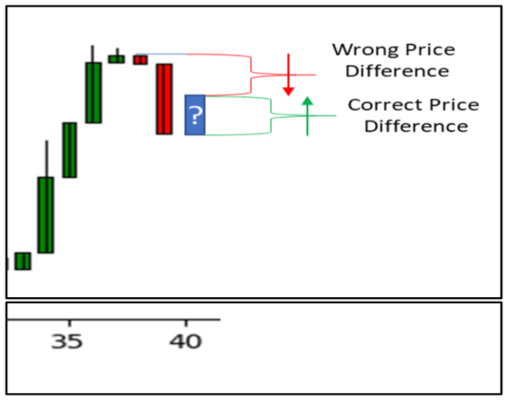

Fig. 4. Image Classification Bug

D. Back Testing Trading Strategy

There are various factors to consider when designing an algorithmic trading bot. The cost of trading (0.1 - 0.25% for most cryptocurrency exchanges), depth of the order-book (there will only be a limited volume to trade at any given price), and speed of APIs all need to be considered when deciding if a strategy will be profitable. The predictive model needs to be transformed into a strategy that can be back tested for historical performance. A simple strategy involving investing testing what would happen if an investor put $1000 into Bitcoin on January 1st, 2018, as shown in Fig. 5. The simple strategy, illustrated above, represents buying when the model predicts upward movement and selling when it predicts downward movement. The model-guided strategy has achieved a 6.6% return on the $1000 investment over the past 2.5 months. This might be not be considered great by some greedy traders who are looking or 1000+% returns (like were experienced in 2017); however, when compared to the "buy-and-hold" strategy, it can be seen just how profitable this model might be. The past 3 months have been considered a bear market for Bitcoin [20]-[24] and a $1000 investment made on January 1st of this year would have lost 35.19%.

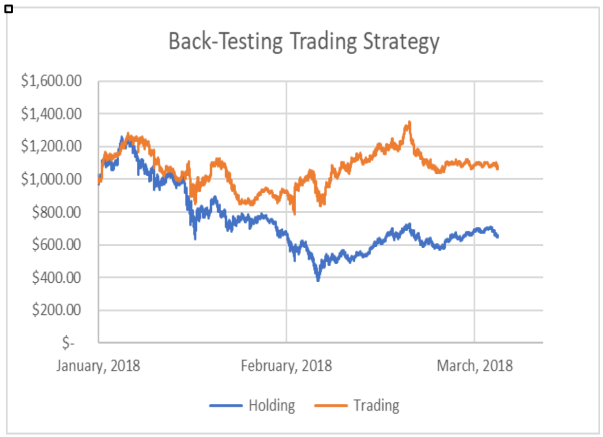

Fig. 5. Back-Testing Strategy

V. Conclusions and Future Directions

The ability to evaluate market movement for a specific cryptocurrency is critical, given the highly volatile and speculative nature of these assets. In this paper, we developed a prediction model using CNN and visual data charts to better predict the movement in Bitcoin prices. The hypothesis is that CNNs are able to identify patterns within image data that humans cannot identify. While the intuition suggests that information is lost when converting structured numerical data into images, new information may be added in the process. Converting data from a 1-dimensional stream into a 2-dimensional image might help "engineer features" that a CNN can detect visually, features that a human would not have thought to create in the 1D data. An accuracy significantly above 70% demonstrates that CNNs can pick up on the patterns within the data, suggesting the validity of the data-chart approach to analyze structured data. Future directions include, moving beyond a simple binary classifier would be the next challenge for this model. Starting with estimating price movement in percentiles, it is possible to extend the CNNs classifier into a full price prediction regression model. Future extension to the work in this paper including, training the whole network on the adequately tagged images.

References

[1] S. Srivastava, Deep Learning in Finance, Towards Data Science, 2017.

[2] S. Razavian, Ali, Azizpour, Hossein, Sullivan, Josephine, Carlsson and Stefan, CNN Features Off-the-Shelf: An Astounding Baseline for Recognition, in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2014.

[3] O. Abdel-Hamid, L. Deng and D. Yu, Exploring Convolutional Neural Network Structures and OptimizationTechniques for Speech Recognition, in INTERSPEECH, Lyon, 2013.

[4] K. Simonyan and A. Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition, in ICLR, 2015.

[5] M. Lin, Q. Chen and S. Yan, Network In Network, CORR Journal, 2014.

[6] M. M. Fischer, Computational Neural Networks — Tools for Spatial Data Analysis, Spatial Analysis and GeoComputation, pp. 79-102, 2006.

[7] E. Silva, D. Castilho, A. Pereira (2014). A neural network-based approach to support the market making strategies in high-frequency trading. IJCNN, 2014 International Joint Conference (pp. 845-852).

[8] Z. Wang, T. Oates (2015). Encoding Time Series as Images for Visual Inspection and Classification Using Tiled Convolutional Neural Networks. Trajectory-Based Behavior Analytics: Papers from the 2015 AAAI Workshop.

[9] K. He, X. Zhang, S. Ren and J. Sun, Deep Residual Learning for Image Recognition, 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 770-778.

[10] I. Loshchilov, F. Hutter, (2017). SGDR: Stochastic Gradient Descent with Warm Restarts, ICLR 2017, 5th International Conference on Learning Representations.

[11] A. Krizhevsky, I. Sutskever, G. Hinton. (2012). ImageNet Classification with Deep Convolutional Neural Networks. Journal of Neural Information Processing Systems.

[12] L.N. Smith, (2017). Cyclical Learning Rates for Training Neural Networks. IEEE Conference on Applications of Computer Vision (WACV), pp. 464-472.

[13] O. Russakosky, O., & J. Deng, (2015). ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision (IJCV), 115(3).

[14] Tiong, L.C., Ngo, D.C., & Lee, Y. (2014). Stock Price Prediction Model using Candlestick Pattern Feature.

[15] P. Rémy, When Bitcoin meets Artificial Intelligence, https://github.com/philipperemy/deep-learning-bitcoin.

[16] X. Tan and R. Kashef. Predicting the closing price of cryptocurrencies: a comparative study. In Proceedings of the Second International Conference on Data Science, E-Learning and Information Systems (DATA '19), 2019.. Association for Computing Machinery, New York, NY, USA, Article 37, 1–5. DOI:https://doi.org/10.1145/3368691.3368728.

[17] D. Shah, (2014). Bayesian regression and Bitcoin. Fifty-second Annual Allerton Conference, USA.

[18] S. Taylor, B. Letham (2017). Prophet: forecasting at scale. Retrieved from Facebook research.

[19] H. Hayashi, (2017). Is Prophet Really Better than ARIMA for Forecasting Time Series Data? Retrieved from Exploratory.

[20] Yermack, D. (2015, May 8). Is Bitcoin a Real Currency? An Economic Appraisal.

[21] Zhengyao Jiang, J. L. (2017). Cryptocurrency Portfolio Management with Deep Reinforcement Learning, Intellisys Conference, London, UK.

[22] https://pytorch.org/

[23] Zhong, X., & Enke, D. (2017). Forecasting daily stock market return using dimensionality reduction. Expert Systems with Applications, pp.126-139.

[24] I. Madan, S. S. (2017). Automated Bitcoin Trading via Machine Learning Algorithms. Semantic Scholar.