Forecasting Cryptocurrency Price Using Convolutional Neural Networks with Weighted and Attentive Memory Channels

Zhuorui Zhang[a], Hong-Ning Dai[a,*], Junhao Zhou[a], Subrota Kumar Mondal[a], Miguel Martínez García[b], Hao Wang[c]

[a]*Faculty of Information Technology, Macau University of Science and Technology, Macau, China*
[b]*School of Aeronautical and Automotive Engineering, Loughborough University, UK*
[c]*Department of Computer Science, Norwegian University of Science and Technology, Norway*

Abstract

After the invention of Bitcoin as well as other blockchain-based peer-to-peer payment systems, the cryptocurrency market has rapidly gained popularity. Consequently, the volatility of the various cryptocurrency prices attracts substantial attention from both investors and researchers. It is a challenging task to forecast the prices of cryptocurrencies due to the non-stationary prices and the stochastic effects in the market. Current cryptocurrency price forecasting models mainly focus on analyzing exogenous factors, such as macro-financial indicators, blockchain information, and social media data – with the aim of improving the prediction accuracy. However, the intrinsic systemic noise, caused by market and political conditions, is complex to interpret. Inspired by the strong correlations among cryptocurrencies and the powerful modelling capability displayed by deep learning techniques, we propose a Weighted & Attentive Memory Channels model to predict the daily close price and the fluctuation of cryptocurrencies. In particular, our proposed model consists of three modules: an Attentive Memory module combines a Gated Recurrent Unit with a self-attention component to establish attentive memory for each input sequence; a Channel-wise Weighting module receives the price of several heavyweight cryptocurrencies and learns their interdependencies by recalibrating the weights for each sequence; and a Convolution & Pooling module extracts local temporal features, thereby improving the generalization ability of the overall model. In order to validate the proposed model, we conduct a battery of experiments. The results show that our proposed scheme achieves state-of-the-art performance and outperforms the baseline models in prediction error, accuracy, and profitability.

Keywords: Cryptocurrency, Time-series forecasting, Convolutional neural networks, Gated recurrent units, Channel weighting, Attention mechanism.

1. Introduction

The world has recently witnessed a rapid growth of the cryptocurrency market. The market capitalization of cryptocurrencies has hit record highs repeatedly. This phenomenon reveals the significant value of cryptocurrencies as an electronic payment system and financial asset (Balcilar et al., 2017). After the invention of Bitcoin (Nakamoto, 2019), one of the most popular cryptocurrencies, the peer to peer payment system has been attracting substantial interest from investors and researchers. Compared with traditional cash systems, cryptocurrencies have unmatched advantages such as decentralization, strong security, and lower transaction charges (Mukhopadhyay et al., 2016). By September 2019, the cryptocurrency market reached a capitalization of 300 billion dollars, with Bitcoin alone accounting for nearly $200 billion. Moreover, more than 2,000 kinds of cryptocurrencies have been launched and are available for public trading.

*Corresponding author

Email addresses: zhuorui.zhang00@gmail.com (Zhuorui Zhang), hndai@ieee.org (Hong-Ning Dai), junhao_zhou@qq.com (Junhao Zhou), skmondal@must.edu.mo (Subrota Kumar Mondal), m.martinez-garcia@lboro.ac.uk (Miguel Martínez García), hawa@ntnu.no (Hao Wang)

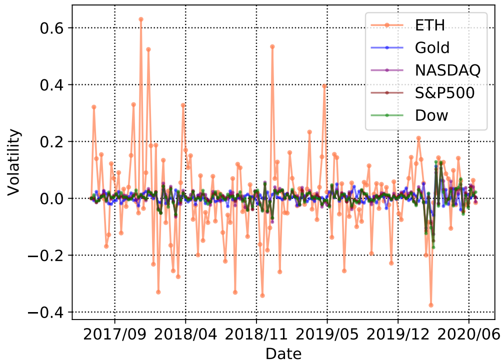


Figure 1: Weekly volatility of ETH compared to that of various classical market indexes.

To make profits and reduce losses, there is a growing demand for developing accurate price-forecasting models for cryptocurrencies. Because the estimation of the price trend in advance may also help in preventing misinformation spread by speculators. However, the prediction task on conventional financial time-series data is notoriously challenging, due to the random characteristics of the market (Kristjanpoller & Minutolo, 2018; Xu & Cohen, 2018). As a type of digital and virtual assets, cryptocurrencies have low correlations with conventional assets, making the analysis of their fluctuations more difficult (Chuen et al., 2017). As an example, we compare the weekly percentage volatility of Ethereum (ETH) with the gold price and other popular financial indexes, including NASDAQ, S&P 500 and DJI. Given weekly data w_i from July 2017 to July 2020, the percentage volatility p_i is computed as $p_i = (w_i - w_{i-1})/w_{i-1} \times 100\%$. It is shown (Figure 1) that the historical volatility of ETH is much higher than those of the classical assets.

Similarly to other assets such as stocks and precious metals, the price of cryptocurrencies is influenced by various factors, e.g., fake news, market manipulation, and government policies. The analysis in (Sovbetov, 2018) recognizes that market beta, trading volume, and volatility are also significant determinants. To tackle the enormous volatility, researchers have developed numerous models which can be categorized into: traditional time-series model such as Autoregressive Integrated Moving Average (ARIMA) and Generalized Autoregressive Conditional Heteroskedasticity (GARCH), and machine learning models – such as Support Vector Machines (SVM) and deep learning nets (Bakar & Rosbi, 2017; Dyhrberg, 2016; McNally et al., 2018; Jang & Lee, 2017). In particular, some approaches use sentiment analysis based on Twitter data, to assist in modelling latent driven factors implicitly (Abraham et al., 2018; Li et al., 2019). Based on analysis on the massive historical data, the Artificial Neural Network (ANN) has powerful feature-representing ability and potentials in forecasting tasks (Esen et al., 2008b,c, 2009). Some approaches also combine the ANN with fuzzy logic to construct adaptive neuro-fuzzy inference systems and show the appropriateness of ANFIS for the quantitative modeling (Esen et al., 2017, 2008a).

However, three main deficiencies of these approaches limit the prediction accuracy: 1) using only a single historical price time-series *ignores important latent driven factors and market information*; 2) it is *complex and impeded* to extract kernel determinants from news sources, such as Twitter and Google trending data due to misinformation and noise; and 3) the impact of various exogenous factors is *difficult to determine and verify* although the results in (Jang & Lee, 2017) suggest that macro-financial markets slightly impact the cryptocurrency prices. In this context, a natural problem arises: what kernel features are worth taking into account, and how to extract those features when forecasting the cryptocurrencies price volatility? Unfortunately, there are few to none systematic analyses of this problem although studies of the correlations among the various cryptocurrencies have been conducted, in which the correlations are examined to be positive (Aslanidis et al., 2019). In particular, some authors investigated the price leadership dynamics of Ethereum and Bitcoin (BTC). The result indicates that it has a lead-lag relationship between them (Sifat et al., 2019). Besides, other researchers confirmed the interdependency between BTC and altcoin markets in short and long periods (Ciaian et al., 2018). Although the interdependencies between cryptocurrencies have been investigated in some articles, few solutions are available towards modelling these.

In this paper, to tackle the above challenges, we propose a Weighted & Attentive Memory Channels (WAMC) model. The WAMC captures the kernel temporal features of cryptocurrency price volatility, and improves the prediction accuracy – by exploiting the correlations among the various cryptocurrency prices. Inspired by the efficacy of GRUs in stock trend prediction (Minh et al., 2018; Vaswani et al., 2017; Radojičić & Kredatus, 2020), we incorporate a GRU component, connected to a self-attention head for each input time-series (the input channels). The self-attention elements yield attentive memory to the model. Moreover, motivated by the Squeeze-and-Excitation Networks in (Hu et al., 2018), which display strong ability for modelling dynamic channel interdependencies, the weights of each channel are recalibrated by way of a Channel-wise Weighting module. Further, a Convolution & Pooling module with regularization is employed, to extract local temporal features and down-sample the data. Different activation functions, such as ReLU (Nair & Hinton, 2010), sigmoid and tanh, have been exploited to achieve non-linear mappings. The main contributions of this paper are summarized as follows:

- The WAMC model is proposed to predict the daily close price and its uptrend and downtrend of cryptocurrencies. The WAMC is efficient in modelling the non-linear correlations between cryptocurrencies. This approach is novel and has not been used in other studies. The WAMC is also capable of extracting local temporal features and establishing attentive memories in short and long periods.
- We investigate the impact of three significant parameters on the WAMC: the window length, the number of convolutional neural network (CNN) layers, and the number of hidden layers in the Channel-wise weighting module. These parameters are shown to influence the training loss and the validation loss in different ways.
- We compare the prediction performance of our proposed model with a battery of econometric, machine learning and deep learning approaches to show its superiority. The WAMC model outperforms all baseline models in commonly used evaluation metrics. In addition, we conduct a number of ablation experiments to examine the significance of each module in the WAMC model.
- We also investigate the effectiveness of the WAMC in capturing and predicting the uptrend and downtrend of the cryptocurrencies price. Compared to other classification models, the WAMC model shows higher accuracy of predicting whether the prices of cryptocurrencies maintain their momentum. Moreover, we exploit the WAMC to take the price fluctuation momentum into consideration and simulate a new investment strategy. We show that the WAMC-based strategy can potentially earn more profits than the strategies only considering the predicted price.

The remainder of the paper is organized as follows. Section 2 presents previous studies on the subject of cryptocurrency price prediction. Section 3 presents the Weighted & Attentive Memory Channels model. Experimental results are given in Section 4. Finally, conclusions are drawn and future work is suggested in Section 5.

2. Related work

In this section, we review recent advances in cryptocurrency price prediction. We roughly categorize existing studies into two types: traditional time-series modelling methods in Section 2.1 and machine learning and deep learning approaches in Section 2.2.

2.1. Traditional time-series modelling

In the financial market, the forecasting of non-stationary time-series financial data such as stock prices has attracted investors and researchers. To address the high stochasticity of both stock and cryptocurrency market, conventional econometric and statistical models have been exploited. Traditional time-series modelling in cryptocurrency price prediction mainly includes the Holt-Winters Exponential Smoothing (Kalekar et al., 2004; Ahmar et al., 2018), ARIMA (Bakar & Rosbi, 2017), GARCH (Katsiampa, 2017) and their variants.

In particular, the ARIMA model used for forecasting bitcoin prices has shown a high prediction accuracy in short time windows (e.g., 2 days) and when the fluctuations are minor. However, when the model is trained in longer periods (e.g., 9 days), the prediction errors were high for longer term prediction. Meanwhile, the study in (Tiwari et al., 2019) compared GARCH with stochastic volatility (SV) models, finding that the SV models are more robust to radical price changes. Moreover, the results showed that the best models for Bitcoin and Litecoin are different for each (i.e., SV-t performs the best for Bitcoin while GARCH-t is the best for Litecoin). In (Chu et al., 2017), the authors trained 12 different GARCH-based models on the log-returns of the exchange rates of seven popular cryptocurrencies, finding that IGARCH and GJRGARCH models fit the volatility best.

Unfortunately, most of these approaches are less accurate in the prediction of highly-fluctuating prices in a long period and also lack a probabilistic interpretation (Amjad & Shah, 2017). In addition, these traditional models are less elastic and flexible when predicting different types of cryptocurrencies.

2.2. Machine learning and deep learning approaches

In contrast to traditional time-series models, machine learning (ML) and deep learning (DL) models have strengths in analyzing non-linear multivariate data with robustness to noise values. Moreover, the prediction of time-series data is essentially related to the regression task in ML and DL methods. Many classical ML models have achieved a higher prediction accuracy compared with traditional econometric and statistical models when sufficient data is available. For example, SVMs, Bayesian Networks (BN), and Multilayer Perceptrons (MLP) have shown promising results of the prediction of cryptocurrency and stock markets (Peng et al., 2018b; Rezaei et al., 2020). In addition, some studies used ML techniques and conducted sentiment analysis on data from social and web search media to investigate the bitcoin volatility (Matta et al., 2015). However, one limitation of these methods is the small sample-size and the influence of misinformation.

As a new developing ML technology, the DL method has made great breakthroughs in the fields of speech recognition, computer vision, and natural language processing. DL is regarded as an effective way to realize time-series prediction. However, studies on predicting the cryptocurrency prices using DL methods are scarce. In recent research, DL is found to be highly efficient in detecting the inherent chaotic dynamics of cryptocurrency markets. In (Lahmiri & Bekiros, 2019), it was found that the predictability of long-short term memory neural network (LSTM) is significantly higher than the generalized regression neural networks (GRNN). Moreover, the work (Dutta et al., 2020) showed that recurrent neural networks (RNN) using gated recurring units (GRU) and LSTM outperform traditional ML models. The GRU model with recurrent dropout significantly improves the performance with respect to the baselines in the prediction of bitcoin prices. In addition, the work (Luo et al., 2019) exploited CNNs to predict the short-term crude oil prices achieving reasonable performance, showing the potential of CNN in predicting cryptocurrencies price. However, these approaches mostly exploit a simple variant of RNN or CNN. Meanwhile, there are few studies on composite DL models in cryptocurrency price prediction. In particular, most of these studies only focus on the price on Bitcoin while ignoring other valuable cryptocurrencies and failing to consider the correlations between cryptocurrencies.

Although (Zhang et al., 2020) developed a Weighted Memory Channels Regression (WMCR) model that learning channel-wise features to improve the forecasting accuracy of cryptocurrency prices, the WAMC model proposed in this paper differs from the previous WMCR model in the following aspects:

- We combine the GRU component with self-attention mechanism to establish the attentive memory in this paper while the previous work (Zhang et al., 2020) only uses LSTM layers to memorize important information and ignore the dependencies of different time slots.

- We also analyze the impact of the depth of hidden layers in the channel-wise weighting module on both training loss and validation loss. Moreover, in this module, we tackle the information loss due to the global average pooling in (Zhang et al., 2020) by replacing it with GRU.

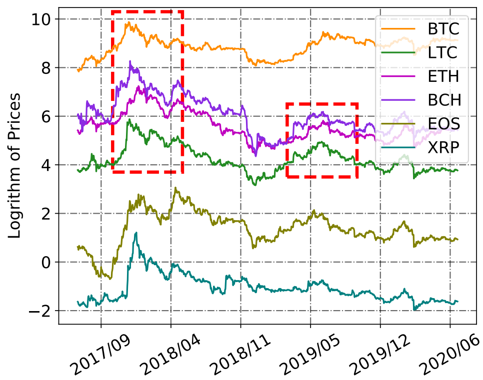


Figure 2: The logarithmic price of different prices.

- We extend our analysis by conducting various ablation studies to examine the significance of each module and how the impact of each module on forecasting errors at different training ratios while these experiments have not been conducted in (Zhang et al., 2020).
- We analyze the generality of WAMC on other cryptocurrencies such as BCH and show the superior performance compared with common baselines. Specifically, we also discuss the parameter settings of baselines in this paper to ensure the sufficient use of them and present a fair comparison with the WAMC.
- We also investigate the investment strategy based on WAMC and conduct extra experiments to evaluate the effectiveness of the WAMC-based strategy with comparison with other classic methods.

3. Our approach

Our approach mainly consists of the following steps: 1) data preprocessing in Section 3.1, 2) Weighted & Attentive Memory Channel model in Section 3.2, 3) training process of the entire model in Section 3.3.

3.1. Data Preprocessing

3.1.1. Correlation analysis of cryptocurrencies

Table 1: The PCC values (in cryptocurrencies pairs)

Cryptocurrencies	BTC	ETH	BCH	LTC	XRP	EOS
BTC	1.00	0.52	0.59	0.69	0.48	0.42
ETH	0.52	1.00	0.88	0.86	0.81	0.75
BCH	0.59	0.88	1.00	0.86	0.80	0.68
LTC	0.69	0.86	0.86	1.00	0.78	0.71
XRP	0.48	0.81	0.80	0.78	1.00	0.68
EOS	0.42	0.75	0.68	0.71	0.68	1.00

We obtain the multi-cryptocurrency price dataset from *CoinMarketCap* at (CoinMarketCap, 2020). To examine the most relevant cryptocurrencies in terms of volatility, we compare the daily closing prices of six popular and valuable cryptocurrencies: BTC, Bitcoin Cash (BCH), Litecoin (LTC), ETH, EOS, and XRP. Figure 2 depicts the fluctuation of logarithmic prices of the six cryptocurrencies. Because the cryptocurrencies were launched at different days, we choose the daily closing prices from July 23, 2017 to July 15, 2020. We observe from Figure 2 that different curves show the similar fluctuation, especially for those in the red boxes. To analyze the correlation degree between different cryptocurrencies, we compute the Person

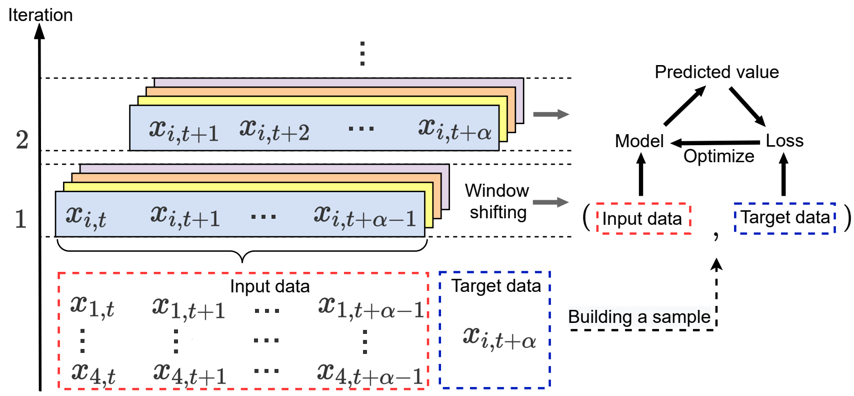


Figure 3: Data preprocessing method.

Correlation Coefficient (PCC) of prices (in pairs). According to the correlation matrix as shown in Tabel 1, PCC values of XRP and EOS corresponding to BTC, ETH, BCH, and LTC are much lower than those of BTC, ETH, BCH, and LTC, implying that XRP and EOS are less relevant to BTC, ETH, BCH, and LTC (i.e., lower PCC means less relevant). In contrast, BTC, ETH, BCH, and LTC have higher PCC values to each other. Therefore, we select BTC, BCH, ETH and LTC to construct the dataset.

3.1.2. Data preprocessing method

In previous studies (Fischer & Krauss, 2018; Bao et al., 2017), price forecasting models were configured by a sliding-window framework. However, each of these models only receives a dataset containing single-price time series data. In this paper, we employ a moving window on multi-channels to generate multi-dimensional inputs. Figure 3 shows the process of data preprocessing and model training. First, we select four cryptocurrencies, each of which has daily closing prices fluctuating in a similar trend. We then use the `StandardScaler` method from scikit-learn (Pedregosa et al., 2011) to standardize the prices. The price of the i-th predicted cryptocurrency ($i \in 1, 2, 3, 4$) in the t-th day is denoted by $c_{i,t}$. We then calculate the standardized price $x_{i,t}$ according to the following equation,

$$x_{i,t} = \frac{c_{i,t} - \overline{c_i}}{\sigma(c_i)}, \tag{1}$$

where $\overline{c_i}$ and $\sigma(c_i)$ are the mean value and the standard deviation of the cryptocurrency price c_i of the i-th cryptocurrency, respectively.

We next employ a moving window with length α on the whole dataset to obtain samples. In each iteration of the sliding process, we obtain a sample consisting of an input data and a target data $x_{i,t+\alpha}$. For example, in iteration 1, the input data contains the standardized closing prices of four cryptocurrencies from day t to day $(t + \alpha - 1)$ and the target data is the standardized price at day $(t + \alpha)$ of the cryptocurrency that we aim to forecast.

After obtaining all samples and selecting a target cryptocurrency to forecast, we feed a batch of samples into our proposed model and compute the loss between the predicted value $\hat{Y}_{t+\alpha}$ and the real value $Y_{t+\alpha}$ at day $(t + \alpha)$. In the training process, we compute the mean squared error as the loss denoted by $\mathbf{L_0}$.

3.2. Weighted & Attentive Memory Channels Model

As illustrated in Figure 4, our WAMC model mainly consists of three modules: the Attentive Memory module, the Channel-wise Weighting module, and the Convolution & Pooling module. We then elaborate these modules as follows.

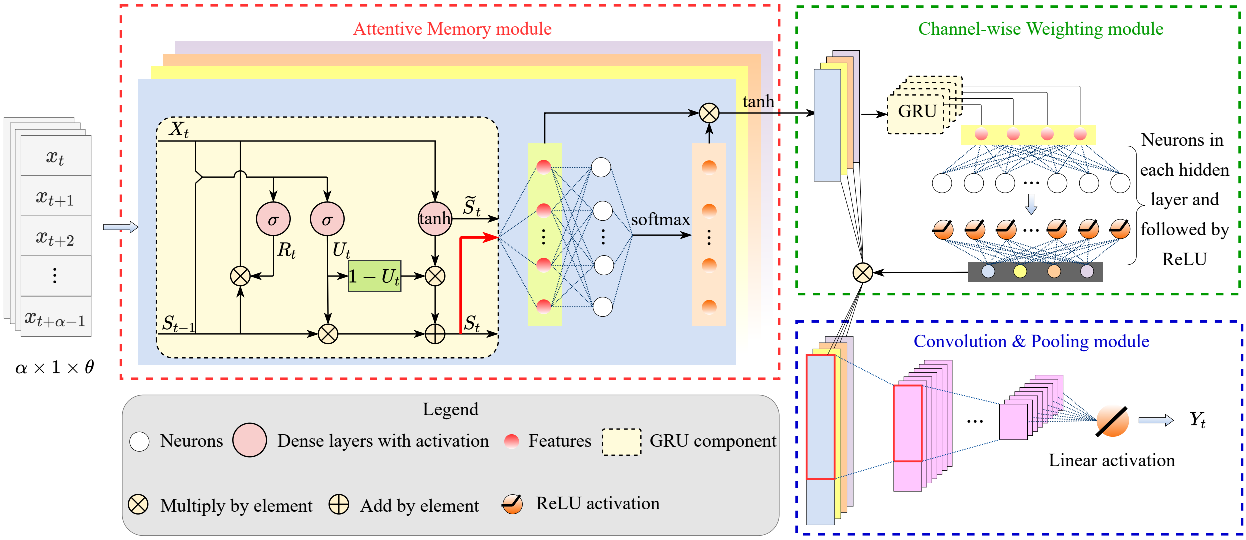


Figure 4: Weighted & Attentive Memory Channels Regression Model.

3.2.1. *Attentive Memory Module*

As shown in Figure 4, the Attentive Memory module compromises two components: a GRU component and a self-attention component. The GRU component uses a reset gate R_t and an update gate U_t, to capture short-term and long-term dependencies in time-series data, respectively. The inputs of both gates are the current time step input X_t and the hidden state of the previous time step S_{t-1}. These two gates are computed as follows:

$$R_t = \sigma(W_{rx}X_t + W_{rs}S_{t-1} + b_r), \tag{2}$$

$$U_t = \sigma(W_{ux}X_t + W_{us}S_{t-1} + b_u), \tag{3}$$

where W_{rx}, W_{rs}, W_{ux} and W_{us} are the trainable parameters while b_r and b_u denote the trainable biases. The activation function logistic sigmoid is represented as σ. Next, the GRU computes the candidate hidden state $\widetilde{S}_t$ to support the later computation of hidden state S_t. The hidden state S_t and the candidate hidden state $\widetilde{S}_t$ are calculated by the following equations, respectively,

$$S_t = U_t \odot S_{t-1} + (1 - U_t) \odot \widetilde{S}_t, \tag{4}$$

$$\widetilde{S}_t = \tanh(W_{sx}X_t + (R_t \odot S_{t-1})W_{ss} + b_s), \tag{5}$$

where $\odot$ represents the Hadamard product.

It is worth mentioning that the closer to 1 an update gate is, the more it suppresses the input of information in previous time steps. On the contrary, when elements in U_t are close to 0, the new hidden state approaches the candidate hidden state. In this manner, GRU tackles the gradient vanishing in RNNs and performs better in capturing dependencies in a long sequence.

Another important component in the Attentive Memory module is the self-attention component. In the GRU component, we encode the input sequence as the hidden state so as to capture the long-term and short-term dependencies. However, the GRU component typically struggles to detect the significance of the instances at different lagged positions. , It is shown in (Liang et al., 2018) that the use of attention mechanisms explicitly enhances the capability of selecting important information at various delayed positions. Therefore, we compute the attention score of each time instance in S_t and use the score to map S_t to an attentive state A_t (i.e., the attentive memory at time t). Given a query q, the attention score a_t of each key-value pair (q, k) is computed by the attention function $f(q, k)$. In particular, the key, value, and query in the self-attention mechanism are equivalent to each other (i.e., the input sequence). In this paper, we employ an MLP with a single hidden layer as the attention function. The i-attentive state at time t denoted by $A_t(i)$ is computed as follows,

$$\begin{aligned} A_t(i) &= \mathrm{softmax}\big(a_t(i)\big) \cdot S_t(i) \\ &= \frac{\exp\big(a_t(i)\big)}{\sum_t \exp\big(a_t(i)\big)} \cdot S_t(i), \end{aligned} \tag{6}$$

where $a_t = f(q,k) = \mathrm{MLP}(S_t)$.

Next, we compute the output O_t of the self-attention component as follows

$$\begin{aligned} O_t(i) &= \big[O_t^{(1)}(i), O_t^{(2)}(i), ..., O_t^{(\alpha)}(i)\big]^T \\ &= \tanh\Big(\big[A_t^{(1)}(i), A_t^{(2)}(i), ..., A_t^{(\alpha)}(i)\big]^T\Big), \end{aligned} \tag{7}$$

where we exploit the activation $\tanh(\cdot)$ to map the data to the interval $[-1,1]$, thereby avoiding the scaling factor $S_t(i)$ by attention weights. Each channel then learns time dependencies to determine which time instances are the more important in the cryptocurrency price series.

3.2.2. Channel-wise Weighting Module

Although the prices of different cryptocurrencies fluctuate in a similar trend, it is still important to determine the importance of distinct sequences. Inspired by the SENet in (Hu et al., 2018), which explicitly models the interdependencies between channels, we propose a Channel-wise Weighting module to recalibrate the significance of different channels (i.e., cryptocurrencies). As shown in Figure 4, the Channel-wise Weighting module exploits a GRU component and multiple dense layers to learn weight vector denoted by ω. Next, the original channel in O_t is multiplied by ω in an element-by-element manner.

Specifically, given θ channels, the Channel-wise weighting module starts with a GRU component (with a single unit) for each channel to generate θ neurons. In the SENet, it uses the global average pooling to generate channel-wise statistics and obtain θ neurons. This aggregation approach is simple and prevalent in image processing. However, SENet ignores time dependencies and correlations between different time slots while the inputs are time-series. Therefore, we adopt a GRU component to tackle this problem. Moreover, because θ neurons are not enough to achieve the flexible connectivity, we then increase the dimensionality in fully-connected layers with β neurons in each layer and followed by the ReLU activation function. The output $s^{(i)}$ of the i-th hidden layer is calculated as follows,

$$\begin{aligned} s^{(i)} &:= \begin{bmatrix} s_1^{(i)} & s_2^{(i)} & \cdots & s_\beta^{(i)} \end{bmatrix} \\ &:= \mathrm{ReLU}(s^{(i-1)}\mathbf{W}), \end{aligned} \tag{8}$$

where the weight parameter denoted by $\mathbf{W}$ can be expressed as follows,

$$\mathbf{W} := \begin{bmatrix} \mathrm{W}_{1,1} & \cdots & \mathrm{W}_{1,\beta} \\ \vdots & \ddots & \vdots \\ \mathrm{W}_{\beta,1} & \cdots & \mathrm{W}_{\beta,\beta} \end{bmatrix}. \tag{9}$$

We also use a dense layer with θ neurons to reduce the dimensionality so as to match the number of channels. The final step in this module is to multiply each channel by the elements in ω to generate weighted channels. The output O_t is computed as follows,

$$O_t = \big[O_t(1), O_t(2), ..., O_t(\theta)\big]^T, \tag{10}$$

where the elements in O_t are computed as in Eq. (7). We denote the output of the Channel-wise Weighting module by O_t^*, which can be calculated as follows,

$$O_t^* = O_t \odot \begin{bmatrix} \omega_1 & ... & \omega_1 \\ \vdots & \ddots & \vdots \\ \omega_\theta & ... & \omega_\theta \end{bmatrix}_{\theta\times\alpha}. \tag{11}$$

Algorithm 1 The learning process of WAMC

Input: Training samples X_t, target samples $Y_{t+\alpha}$, window length α

Output: Price prediction $\hat{Y}_{t+\alpha}$ of the target cryptocurrency

for each epoch **do**
    for each step in an epoch **do**
        randomly select a batch of samples X_t^b
        1. Attentive memory module
        GRU: establish a preliminary memory
        Self-attention: compute the attentive memory O_t^b
        2. Channel-wise weighting module
        Generate a global hidden state for each channel
        Compute the weights vector ω
        Obtain weighted & attentive memory channels
        3. Convolution & Pooling module
        Extract local features by convolution layers
        Down-sampling by pooling layers
        Connect all feature maps and output the $\hat{Y}_{t+\alpha}$
    end for
    Calculate the loss between $Y_{t+\alpha}$ and $\hat{Y}_{t+\alpha}$ in Eq. (14)
    Optimize the parameters by Adam optimizer
end for

3.2.3. Convolution & Pooling Module

Although Artificial Neural Networks (ANN) and Recurrent Neural Networks (RNN) can learn complex nonlinear mappings and capture time dependencies, the full connectivity of neurons limits the flexibility and scalability of the network. Compared with conventional ANN and RNN, CNN connects each neuron with the adjacent neurons by locally sharing weights, and identifying significant local features more efficiently. The group of neighbouring neurons can also be regarded as a local receptive field. In CNN, the convolutional layers extract local features and the pooling layers to reduce the parameter number and minimize the over-fitting effect. We therefore employ the 2-D convolutional layer with the max-pooling layer in this module.

More specifically, the first convolutional layer in this module receives the O_t^* as the input and performs a 2-D convolution for each channel. Each convolutional layer having 64 filters is activated by ReLUs. We define the convolution as Conv2D$(\cdot)$. The output $\mathrm{Y}_{\mathrm{conv}}$ of the first convolutional layer is then expressed as follows,

$$\mathrm{Y}_{\mathrm{conv}} = \mathrm{ReLU}\big(\mathrm{Conv2D}(w^*, O_t^*) + b^*\big), \tag{12}$$

where w^* and b^* denote the weights and the biases, respectively.

Moreover, we use an ℓ_2-regularization and an ℓ_1-regularization to enforce overfitting penalties on w^* and b^*, respectively. As the most widely used regularization methods, ℓ_1 and ℓ_2 can improve the generalization ability and prevent over-fitting. The original loss function $\mathbf{L_0}$ is given by,

$$\mathbf{L_0} = \frac{1}{N}\sum_{i=1}^{N}(\hat{Y}_i - Y_i)^2, \tag{13}$$

where N denotes the number of predicted values, $\hat{Y}_i$ and Y_i represent the predicted value and the real value, respectively. The loss function with regularization is defined as $\mathbf{L}$, which is given as follows,

$$\mathbf{L} = \mathbf{L}_0 + \lambda\Big(\sum_{i=1}^{n}\sum_{j=1}^{m}|w_{ij}^*|^2 + \sum_{i=1}^{N}|b_i^*|\Big), \tag{14}$$

where the weight parameter in the convolution kernel w^* is denoted by w_{ij}^*, b_i^* denotes the bias term in b^*, and $|\cdot|$ takes the absolute value. In the WAMC model, we first vary λ at different magnitude orders (e.g.,

$\lambda \in \{0.001, 0.01, 0.1\}$. According to the validation loss of the WAMC, the WAMC performs the best when λ is on the order of 0.01. However, the improvement due to the varied λ around 0.01 (e.g., 0.02) on the performance is not significant, we therefore fix λ to be 0.01 in the WAMC. Next, a max-pooling layer is employed to perform down-sampling. In the max-pooling layer, it only retains the maximum value in the area scanned by the pooling filter. Finally, we connect all feature maps generated by the last pooling layer together and output the predicted value $\hat{Y}_{t+\alpha}$.

3.3. Training Process

The main training procedure of WAMC is summarized in Algorithm 1. We adopt four types of cryptocurrency as the four input channels in each training sample – and select one of them as the target to predict. In the training process, large batch sizes b lead to poor generalization and convergence to local optimum. Meanwhile, small b causes a slower convergence and costs more training time in an epoch (Hoffer et al., 2017). To tackle this trade-off, we test various values of b, and select $b = 100$ as an effective compromise. In particular, the three modules in Algorithm 1 are responsible for improving the prediction accuracy and the generalization ability of the model. The attentive memory module establishes attentive memory for each channel (line 4-6). The channel-wise weighting module discriminates the significance of each channel by learning a weights vector (line 7-10). The convolution & pooling module extracts the local features and applies a regularization on all the channels (line 11-14). Finally, we calculate the loss as in Eq. (14) and optimize the parameters by Adam optimizer in the process of backpropagation (line 16-17).

4. Experiments

4.1. Experimental settings

We conduct experiments to evaluate the proposed WAMC. We obtain cryptocurrency datasets from *CoinMarketCap* at (CoinMarketCap, 2020) and preprocess datasets as described in Section 3.1.1. Finally, we obtain a dataset containing 1,089 samples.

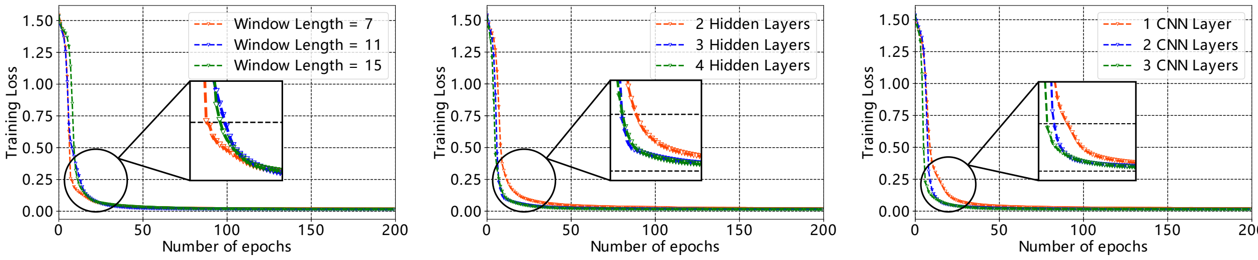
4.1.1. Training settings

We perform the experiments with Keras (Tensorflow as backend) and use Adam with the learning rate of 0.001 to optimize all the models. Regarding parameter study in Section 4.2, we divide the dataset into the training subset (60% samples), the validation subset (20% samples), and the test subset (20% samples). Regarding performance comparison in Section 4.3, we redivide the dataset into the training subset (80% samples) and the validation subset (20% samples). Meanwhile, each sample is essentially a 3-dimensional tensor with the dimensions arranged as $\alpha \times 1 \times \theta$ ($\alpha = 7$, $\theta = 4$). Moreover, according to the results of parameter study, both training loss and validation loss mostly converge after 50 training epochs and stabilize after 100 epochs. In view of efficiency and feasibility, we train models 100 epochs with 10 steps per epoch. Before every training process, a random seed is fixed to ensure that the results are reproducible.

4.1.2. Evaluation metrics

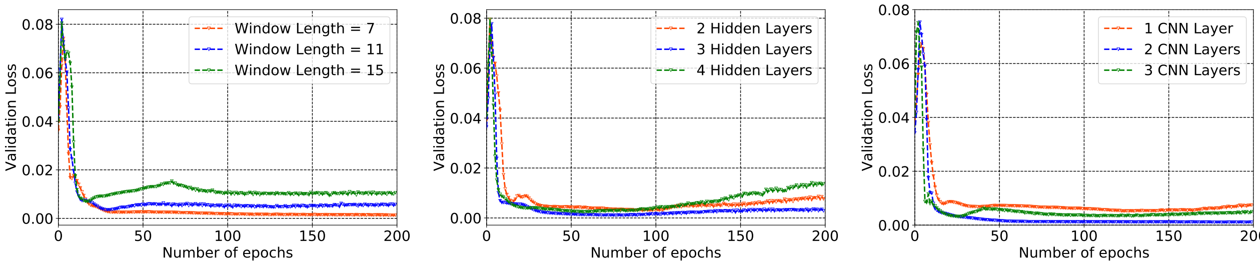
We adopt four standard metrics to evaluate the prediction performance: the root mean square error (RMSE), the mean absolute error (MAE), and the mean absolute percentage error (MAPE) to evaluate the prediction error while R-squared (R^2) to evaluate accuracy. A brief introduction to these four metrics is given as follows.

RMSE takes the square root of the average of summed squared errors between predicted value $\hat{Y}_i$ and real value Y_i. The equivalent formula for RMSE is $\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{Y}_i - Y_i)^2}$, where N denotes the number of predicted values. MAE represents the mean absolute error between Y_i and $\hat{Y}_i$, being expressed as $\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N}|\hat{Y}_i - Y_i|$. MAPE computes the accuracy as a ratio (i.e., the average of the ratio of the residual to the actual value), being defined as $\mathrm{MAPE} = \frac{100\%}{N}\sum_{i=1}^{N}\left|\frac{\hat{Y}_i - Y_i}{Y_i}\right|$. The metric R^2 evaluating the fitness of the regression model is calculated as $\mathrm{R}^2 = 1 - \sum_{i=1}^{N}(Y_i - \hat{Y}_i)^2 / \sum_{i=1}^{N}(Y_i - \bar{Y}_i)^2$. An R^2 closer to 1 indicates a better regression fitting.



(a) Effect of α (b) Effect of β (c) Effect of γ

Figure 5: The impact of the significant parameters on training loss



(a) Effect of α (b) Effect of β (c) Effect of γ

Figure 6: Impact of the significant parameters on the validation loss.

4.2. Parameter study

We investigate the effect of various parameters on the performance of WAMC. To eliminate the influence caused by prediction of different cryptocurrencies, we consider a single cryptocurrency (i.e., the ETH). We compute the Mean Squared Error (i.e., the square of RMSE) to evaluate the prediction loss on both training set and validation set. Table 2 summarizes the training loss and validation loss after 200 epochs of training.

Table 2: Influence of different parameters

Evaluation metrics	Window length α			No. of hidden layers β			No. of CNN Layers γ		
	7	**11**	**15**	**2**	**3**	**4**	**1**	**2**	**3**
Training loss	0.0179	**0.0140**	0.0145	0.0180	0.0143	**0.0137**	0.0179	0.0136	**0.0133**
Validation loss	**0.0013**	0.0057	0.0105	0.0076	**0.0028**	0.0137	0.0074	**0.0011**	0.0050

4.2.1. Impact of window length α

We first analyze the effect of window length α. It is a parameter that controls the length of each channel in a sample. A bigger α indicates that each sample feeds more historical prices into the model. We vary α from 7, 11 to 15. Meanwhile, the parameters β and γ are fixed to 3 and 2, respectively.

Figure 5(a) and Figure 6(a) show the training loss and validation loss in 200 epochs, respectively. As shown in Figure 5(a), the model with a smaller α shows a faster convergence. The faster convergence rate may owe to a shorter time step (i.e., α) in the GRU component is easier to train. Though the training losses all stabilize at a similar level after 100 epochs, the validation loss is evidently impacted by α as shown in Figure 6(a). There is an increment of the validation loss with the rising of α from 7, 11 to 15. Since the cryptocurrency prices that sharply fluctuate are little affected by historical prices more than a week ago, a higher α may impair the memory establishing process.

4.2.2. Impact of hidden layers β in channel-weighting module

In the channel-weighting module, the number of hidden layers denoted by β influences the depth of the network and the ability to learn an appropriate weight vector. We vary β from 2 to 4 with the step value of 1 while fixing $\alpha = 7$ and $\gamma = 2$. As shown in Figure 5(b) and Table 2, the training loss of the WAMC with $\beta = 2$ converges slower than others, indicating a poorer learning ability. Moreover, when β increases from 2 to 3, the validation loss stabilizes at a lower value. However, when β increases to 4, the validation loss rises after approximately 100 epochs of training. This result suggests that $\beta = 3$ is a better choice while a smaller β lacks fitting ability and a higher value of β causes over-fitting.

4.2.3. Impact of CNN layers γ

We also consider the effect of the number of CNN layers denoted by γ. We vary γ from 1, 2 to 3. At the same time, we fix α to be 7 and fix β to be 3. We employ a Max-pooling layer after each CNN layer to conduct down-sampling. Figure 5(c) shows the training loss of the model with different values of γ. It can be seen that the model with a single CNN layer demands more epochs to achieve the lowest loss. We suspect that a single CNN layer lacks the ability to extract spatial features among adjacent data. Moreover, as shown in Figure 6(c), when γ increases from 1 to 2, the validation loss reduces significantly. However, the validation loss rebounds slightly when γ increases to 3. This result implies that $\gamma = 2$ is an optimal value for WAMC and fewer CNN layers are struggling to extract features. Moreover, more CNN layers that have more parameters do not help to reduce prediction errors. Since we use the prices in a short window as the input (i.e., $\alpha = 7$), this result indicates that a deeper CNN with more than 2 layers is not necessary and may hurt the generalization.

4.3. Performance comparison

We then conduct the experiments on ETH dataset and BCH dataset and compare the performance of the WAMC model with the following conventional baselines.

- **Support Vector Regression (SVR)** is a significant application of SVM in regression tasks. Due to its advantages of recognizing patterns of small, non-linear, and high-dimensional samples, previous studies mainly exploit SVR to forecast financial data.
- **ARIMA** is one of the most popular statistical models used for time-series prediction.
- **Random Forest Regressor (RF-Regressor)** is an ensemble learning model for regression tasks.
- **XGBoost Regressor (XGB-Regressor)** is an improved implementation of Gradient Boosted Decision Trees (GBDT) with better performance than GBDT.
- **MLP** overcomes the weakness that the perceptron cannot recognize the linear indivisible data. Combined with non-linear activation functions, it can solve complex problems in many fields including speech and image recognition.
- **LSTM** was adopted to tackle the problem of gradient vanishing explosion in long-sequence training in (Hochreiter & Schmidhuber, 1997). By exploiting three gates: input gate, output gate and forget gate, the LSTM neural networks filter information selectively.
- **GRU** replaces the input gate and forget gate in LSTM with a single update gate. It performs similarly to LSTM, but has a simpler architecture and fewer parameters.
- **CNN** mainly consists of convolution layers, pooling layers and fully-connected layers. This architecture brings CNN the ability of representation learning and effectively alleviates the over-fitting.
- **LSTM+CNN** is a hybrid structure of LSTM and CNN. It is capable of capturing long-term dependency and extracting short-term features.

Table 3: Kernel parameters of baseline models

Models	**Parameter settings**
ARIMA	(p, d, q): $(0, 1, 0)$ for ETH, $(2, 1, 4)$ for BCH
RF-Regressor	number of trees: 100
XGB-Regressor	number of trees: 100
SVR	regularization parameter: 0.1
	kernel: radial basis function (RBF)
MLP	number of layers: 4
	number of neurons in each layer: 32
LSTM	number of layers: 2
	number of hidden units: 7
GRU	number of layers: 2
	others: same as GRU component of WAMC
CNN	same as Convolution & Pooling module of WAMC

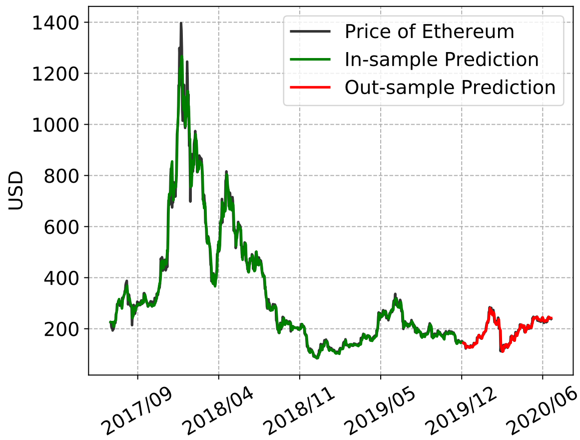
- **GRU+CNN** is also a hybrid structure that uses GRU as the RNN component. Compared with LSTM+CNN, this baseline captures both spatial and temporal features while having low computing cost.

Table 3 summarizes the key parameters of each baseline. It is worth mentioning that in the DL-based baselines, there are various common parameters such as the number of neurons, number of layers and activation functions. To make the comparison relatively fair, we make these common parameters retain consistency with the parameters of the WAMC model. For example, in CNN, the parameters such as the number of filters, the number of layers and the kernel size are the same as those in the Convolution & Pooling module of WAMC. For the sake of simplicity, the parameters of the remaining baselines are not shown.

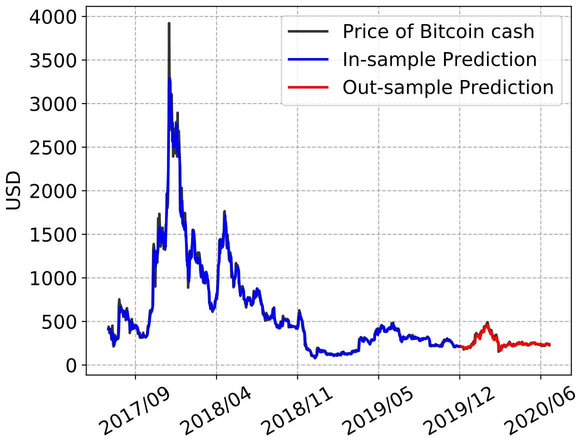
Table 4: Performance comparison

Models	Ethereum				Bitcoin cash			
	RMSE	MAE	MAPE	R-squared	RMSE	MAE	MAPE	R-squared
ARIMA	6.36e+01	5.24e+01	2.46e+01	−1.05e+00	9.05e+01	6.18e+01	1.95e+01	−6.55e-01
SVR	1.56e+01	1.16e+01	6.37e+00	8.76e-01	2.64e+01	2.07e+01	7.69e+00	8.60e-01
RF-Regressor	1.38e+01	9.88e+00	5.34e+00	9.03e-01	2.13e+01	1.55e+01	5.99e+00	9.09e-01
XGB-Regressor	1.72e+01	1.42e+01	8.35e+00	8.49e-01	2.21e+01	1.64e+01	6.68e+00	9.01e-01
MLP	2.06e+01	1.40e+01	7.79e+00	7.84e-01	2.20e+01	1.69e+01	8.93e+00	7.56e-01
LSTM	2.04e+01	1.63e+01	9.35e+00	7.89e-01	2.81e+01	1.95e+01	7.25e+00	8.40e-01
GRU	1.76e+01	1.35e+01	7.73e+00	8.43e-01	2.60e+01	1.92e+01	7.59e+00	8.64e-01
CNN	1.44e+01	1.02e+01	5.45e+00	8.94e-01	2.12e+01	1.53e+01	6.16e+00	9.09e-01
LSTM+CNN	1.37e+01	9.30e+00	5.17e+00	9.06e-01	2.06e+01	1.35e+01	5.06e+00	9.15e-01
GRU+CNN	1.21e+01	8.47e+00	4.74e+00	9.26e-01	1.93e+01	1.33e+01	5.06e+00	9.25e-01
WAMC	**9.70e+00**	**5.97e+00**	**3.28e+00**	**9.52e-01**	**1.73e+01**	**1.15e+01**	**4.38e+00**	**9.40e-01**

As shown in Table 4, our WAMC model achieves the lowest prediction errors (i.e., RMSE, MAE, and MAPE) and the highest prediction accuracy (i.e., the R-squared) on both datasets. For example, the WAMC achieves the lowest RMSE of 9.70 and achieves the maximum R-squared of 0.952 on the test set of the ETH dataset. Meanwhile, the traditional parametric model: ARIMA shows the worst performance. Among the solo-structured ML and DL baselines, CNN and RF-Regressor outperform others. In particular, the performance is improved in the composite DL models: LSTM+CNN and GRU+CNN. This improvement may owe to the integration of the memory establishing ability of RNN components and spatial features extracting ability of CNN. Moreover, GRU+CNN performs the best among all baselines. The superiority of GRU+CNN also implies the advantages of using GRU as the RNN component and exploiting a Convolutional



(a) Prediction of ETH



(b) Prediction of BCH

Figure 7: Prediction of the WAMC on ETH and BCH.

& Pooling module to extract features in our WAMC scheme.

Figure 7 shows the in-sample and out-sample prediction of WAMC on the price of two cryptocurrencies: ETH and BCH. It is shown that our proposed model can accurately forecast the prices of different cryptocurrencies.

Table 5: Ablation analysis of WAMC at different training ratios

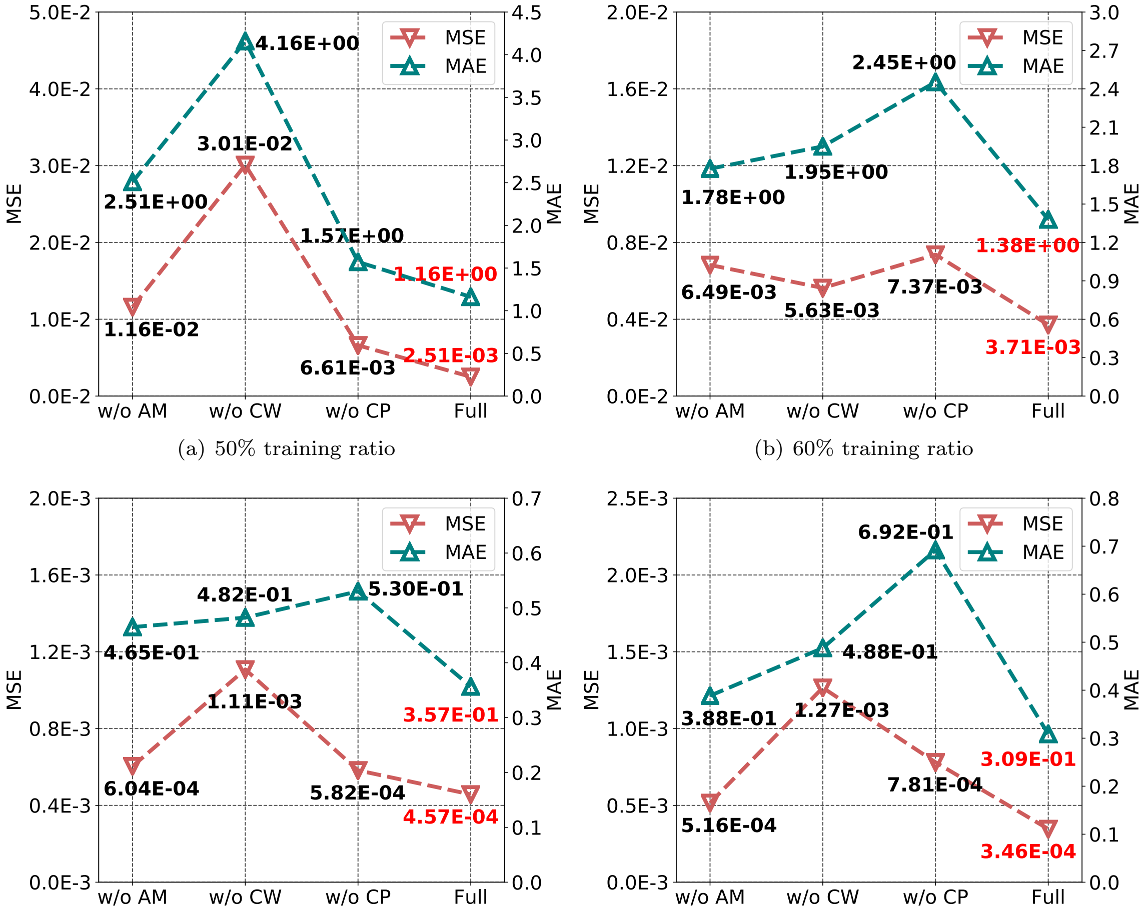
Models	MSE				MAE			
	50%	60%	70%	80%	50%	60%	70%	80%
w/o AM	1.16e-02	6.49e-03	6.04e-04	5.16e-04	2.51e+00	1.78e+00	4.65e-01	3.88e-01
w/o CW	3.01e-02	5.63e-03	1.11e-03	1.27e-03	4.16e+00	1.95e+00	4.82e-01	4.88e-01
w/o CP	6.61e-03	7.37e-03	5.82e-04	7.81e-04	1.57e+00	2.45e+00	5.30e-01	6.92e-01
Full WAMC	**2.51e-03**	**3.71e-03**	**4.57e-04**	**3.46e-04**	**1.16e+00**	**1.38e+00**	**3.57e-01**	**3.09e-01**

4.4. Ablation experiments

We make an elaborative analysis of three principal modules within our WAMC scheme. These modules are the Attentive Memory module, the Channel-wise Weighting module, and the Convolution & Pooling module. To investigate the significance of each module, we construct the following variations of WAMC:

- WAMC (w/o AM) is a variant of the full WAMC without Attentive Memory module. In particular, we replace the GRU component in the module by a fully-connected layer to maintain the depth of the network.
- WAMC (w/o CW) is a variant of the full WAMC without Channel-wise Weighting module. In this model, the outputs of the Attentive Memory module are directly fed into the Convolution & Pooling module.
- WAMC (w/o CP) is a variant of the full WAMC without Convolution & Pooling module. In particular, we replace the two CNN layers in the module by two dense layers to maintain the depth of the network. To ensure fairness, we keep the same parameters such as the number of neurons and the activation function as the variants of WAMC (i.e., before and after the replacement).

We conduct the ablation study on the ETH dataset and evaluate the prediction errors (MSE and MAE) on the test set. The experiment results are shown in Figure 8 and Table 5. With four different training



(a) 50% training ratio (b) 60% training ratio

(c) 70% training ratio (d) 80% training ratio

Figure 8: Sensitivity analysis of the parameters, showcasing their impact on training loss.

ratios being 50%, 60%, 70% and 80%, the impacts of eliminating different modules on prediction errors are varied. At the same time, compared to the full WAMC, it can be noticed that removing any primary module would deteriorate the performance (i.e., increasing MSE or MAE). Specifically, removing the Channel-wise Weighting module at a low training ratio (i.e., 50%) significantly increases both MSE and MAE. This increment indicates the advantages of rescaling different channels of cryptocurrencies while predicting one of them. Furthermore, the Convolution & Pooling module plays an important role in generalizing features due to its ability to capture features in adjacent data points. Meanwhile, compared with other modules, the Attentive memory module contributes to a relatively slight but also remarkable improvement on the performance of the full WAMC. This relatively low contribution to the performance may attribute to the high volatility of prices, thereby making the historical prices less related to the present prices and leading to a hard memory establishing. According to the ablation results in Table 5, each proposed module contributes to the prediction accuracy at different degrees. The completed WAMC scheme with integrated modules has the strongest capabilities of feature learning and generalization.

4.5. The price momentum prediction and investment strategies

Besides forecasting the cryptocurrencies price, it is also worth investigating the fluctuation of *price momentum* so that we can design investment strategies for executing more profitable trades. A simple trading strategy is: buy new cryptocurrencies if the price is predicted to rise and sell them if the price is

Table 6: Experimental results on Ethereum market

Models	Accuracy %	Precision %	Recall %	F1 %	AUC	Profit	Return %
$\{\hat{p_t}\}$	-	-	-	-	-	183.10	7.47
Bagging	68.35	66.95	72.48	69.60	0.7375	196.46	8.29
RF	66.51	66.07	67.89	66.97	0.7359	184.20	8.01
SVM	70.64	69.91	72.48	71.17	0.7860	189.42	8.03
XGB	69.27	69.44	68.81	69.12	0.7508	181.56	8.40
GRU	72.94	72.32	74.31	73.30	0.8215	196.88	7.59
LSTM	72.48	71.30	75.23	73.21	0.8198	206.38	7.98
WAMC	**77.52**	**76.58**	**78.70**	**77.63**	**0.8234**	**216.93**	**8.86**

predicted to decline. However, the profits made by this investment strategy may be significantly reduced because of the regression errors. Meanwhile, it is unrewarding to execute frequent short-term tradings, especially for slight fluctuations that are usually imprecise and difficult to be distinguished from prediction errors.

Table 7: Experimental results on Bitcoin cash market

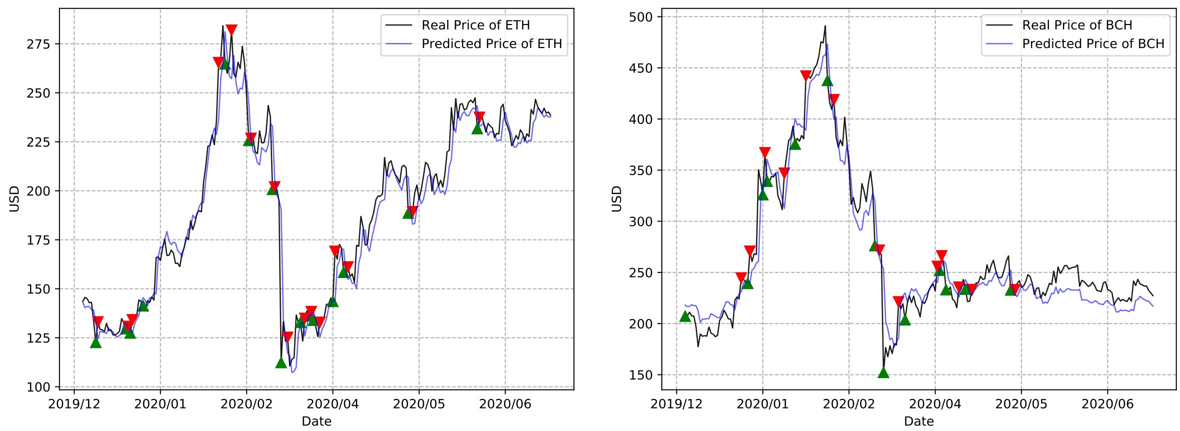
Models	Accuracy %	Precision %	Recall %	F1 %	AUC	Profit	Return %
$\{\hat{p_t}\}$	-	-	-	-	-	267.69	8.22
Bagging	70.64	68.70	73.83	71.17	0.7856	273.33	7.64
RF	71.56	71.43	70.09	70.75	0.7995	272.80	7.21
SVM	70.64	65.47	**85.05**	73.98	0.7740	261.09	6.90
XGB	72.94	72.64	71.96	72.30	0.7983	270.10	7.69
GRU	71.10	69.30	73.83	71.49	0.7941	276.75	7.33
LSTM	71.56	70.27	72.90	71.56	**0.8156**	270.62	7.15
WAMC	**75.23**	**74.31**	**75.70**	**75.00**	0.8060	**299.11**	**9.06**

To tackle this issue, we first investigate the volatility prediction of the price increment and decrement. For instance, when the price is predicted to maintain its momentum with high confidence, increasing the investment is a profitable choice. Let p_t and $\hat{p_t}$ represent the real price and predicted price of a cryptocurrency at time $t \in \mathbb{Z}^+$, respectively. The price variance (increment or decrement) is defined as $d_t = p_t - p_{t-1}$ (being regarded as the price momentum). At any time t, an investment strategy can be described as: decide whether to buy or sell cryptocurrencies according to the price prediction and the current investment. To demonstrate the utility of accurate predictions and conduct comparisons, we consider a simple investment strategy that ignores transaction costs and the effect of trading volume. We consider that the cryptocurrency can simultaneously be possessed by one and h_t represents the number of the cryptocurrencies being possessed at time t. At time t, we label it as a positive when $d_t \geq d_{t-1}$ and label it as a negative when $d_t < d_{t-1}$. In the classification, the last activation of the WAMC is replaced by the sigmoid function. The input of the model is the price fluctuation in every α days (i.e., $[d_{t-\alpha+1} : d_t]$ while the output o_t is the probability of being a positive at time $t+1$ (i.e., the probability that the price maintains its momentum). Given $\hat{p_t}$ and $o_t \in [0, 1]$, the action a_t is defined by:

$$a_t = \begin{cases} \text{buy}, & \text{if } h_t = 0 \ \& \ o_t > 0.5 \ \& \ \hat{p_t} > p_{t-1}; \\ \text{sell}, & \text{if } h_t = 1 \ \& \ o_t < 0.5 \ \& \ \hat{p_t} < p_{t-1}; \\ \text{hold}, & \text{otherwise.} \end{cases}$$

Compared to the strategies that only consider whether the price will rise or fall, we also consider whether the price will maintain its uptrend or downtrend. For instance, at time t, we decide to buy cryptocurrencies while the price is predicted to rise ($\hat{p_{t+1}} > p_t$) and to maintain its uptrend ($d_{t+1} \geq d_t$).

To evaluate the effectiveness of WAMC in predicting whether the uptrend and downtrend will continue or discontinue, we compare the WAMC-based strategy with classic methods such as Bagging, Random Forests (RF) (Patel et al., 2015), Support Vector Machine with radial basis function kernel (SVM) (Peng et al.,



(a) Investment on ETH

(b) Investment on BCH

Figure 9: Investment strategy of the WAMC on ETH and BCH.

2018a) and XGBoost (XGB) (Chen & Guestrin, 2016). Meanwhile, the GRU and LSTM for classification have also been evaluated. In particular, we use the predicted price $\{\hat{p_t}\}$ given by the WAMC (for regression) in Section 4.3 considering its high prediction accuracy. When training the classification model to predict whether d_t will increase or decrease, we compute the binary cross-entropy as the loss function. Other parameters such as the training ratio, learning rate and optimizer are the same as those in Section 4.3.

Besides computing the evaluation metrics for classification, we also compute the cumulative profits and returns on investments when using the above mentioned investment strategies. The experimental results are shown in Table 6 and Table 7. We observe that the WAMC shows high prediction accuracy of the volatility of d_t on both the ETH market and the BCH market. Therefore, our WAMC can make higher profits and achieve higher returns. This superiority may attribute to the great feature-capturing capability of the cryptocurrency-price volatility of the WAMC. In particular, when only considering the predicted prices $\{\hat{p_t}\}$ to make investments, the profits and returns (i.e., profit=183.10, return=7.75 on the ETH test set; profit=267.69, return=8.22 on the BCH test set) are lower than the strategies that also consider the accurate prediction of the volatility of d_t. Fig. 9 further illustrates the investments of the WAMC on the test sets of ETH and BCH markets (the green triangular markers represent buying and the red triangular markers represent selling). It can be noticed that the WAMC shows the effectiveness in buying low and selling high.

5. Conclusion

In this paper, we have proposed a Weighted & Attentive Memory Channels model to predict the daily close price and the fluctuation of cryptocurrencies. Specifically, our Weighted & Attentive Memory Channels model is composed of an Attentive Memory module, a Channel-wise Weighting module and a Convolution & Pooling module. In our proposed model, the processes of attentive memory establishing, channel-wise recalibrating, and convolution and pooling are independently consequently improving the ability of modelling complex dynamics, such as those in the cryptocurrency market. We also have conducted extensive experiments using historical price data from four heavyweight cryptocurrencies. For the price regression, the experimental results show the higher efficacy of our proposed model with respect the lower prediction errors and higher accuracy, as compared to other prevailing methods. Moreover, we also conduct experiments to investigate parameter sensitivity. The results show that our proposed model only requires a short time window and few layers (e.g., 2 CNN layers and 3 hidden layers) to achieve high prediction accuracy. This may attribute to the feature learning of each module from different aspects such as time dependencies capturing and cryptocurrency-channels recalibrating. Meanwhile, the ablation study validated that each module contributes to reducing the forecasting errors evidently with various training ratios. The WAMC model also shows a promising accuracy for predicting whether the price will increase or reduce its momentum (e.g., will the price increase more or increase less?). Compared to those only caring about the predicted price, the consideration of the price momentum may help the model select more reasonable opportunities of buying and selling. Future work will be directed towards examining the applicability of the Weighted & Attentive Memory Channels model in other financial markets, such as stock price forecasting, and to further investigate the adaptability and flexibility of the model with respect to different cryptocurrencies.

References

Abraham, J., Higdon, D., Nelson, J., & Ibarra, J. (2018). Cryptocurrency price prediction using tweet volumes and sentiment analysis. *SMU Data Science Review*, *1*, 1.

Ahmar, A., Rahman, A., & Mulbar, U. (2018). α-sutte indicator: a new method for time series forecasting. In *J. Phys. Conf. Ser* (p. 012018). volume 1040.

Amjad, M., & Shah, D. (2017). Trading bitcoin and online time series prediction. In *NIPS 2016 Time Series Workshop* (pp. 1–15).

Aslanidis, N., Bariviera, A. F., & Martínez-Ibañez, O. (2019). An analysis of cryptocurrencies conditional cross correlations. *Finance Research Letters*, *31*, 130–137.

Bakar, N. A., & Rosbi, S. (2017). Autoregressive integrated moving average (arima) model for forecasting cryptocurrency exchange rate in high volatility environment: A new insight of bitcoin transaction. *International Journal of Advanced Engineering Research and Science*, *4*, 237311.

Balcilar, M., Bouri, E., Gupta, R., & Roubaud, D. (2017). Can volume predict bitcoin returns and volatility? a quantiles-based approach. *Economic Modelling*, *64*, 74–81.

Bao, W., Yue, J., & Rao, Y. (2017). A deep learning framework for financial time series using stacked autoencoders and long-short term memory. *PloS one*, *12*, e0180944.

Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* KDD '16 (p. 785–794). New York, NY, USA: Association for Computing Machinery. URL: `https://doi.org/10.1145/2939672.2939785`. doi:`10.1145/2939672.2939785`.

Chu, J., Chan, S., Nadarajah, S., & Osterrieder, J. (2017). Garch modelling of cryptocurrencies. *Journal of Risk and Financial Management*, *10*, 17.

Chuen, D. L. K., Guo, L., & Wang, Y. (2017). Cryptocurrency: A new investment opportunity? *The Journal of Alternative Investments*, *20*, 16–40.

Ciaian, P., Rajcaniova, M. et al. (2018). Virtual relationships: Short-and long-run evidence from bitcoin and altcoin markets. *Journal of International Financial Markets, Institutions and Money*, *52*, 173–195.

CoinMarketCap (2020). Coinmarketcap cryptocurrency price. `https://coinmarketcap.com/`. Accessed: 2020-07-20.

Dutta, A., Kumar, S., & Basu, M. (2020). A gated recurrent unit approach to bitcoin price prediction. *Journal of Risk and Financial Management*, *13*, 23.

Dyhrberg, A. H. (2016). Bitcoin, gold and the dollar–a garch volatility analysis. *Finance Research Letters*, *16*, 85–92.

Esen, H., Esen, M., & Ozsolak, O. (2017). Modelling and experimental performance analysis of solar-assisted ground source heat pump system. *Journal of Experimental & Theoretical Artificial Intelligence*, *29*, 1–17. URL: `https://doi.org/10.1080/0952813X.2015.1056242`. doi:`10.1080/0952813X.2015.1056242`. `arXiv:https://doi.org/10.1080/0952813X.2015.1056242`.

Esen, H., Inalli, M., Sengur, A., & Esen, M. (2008a). Artificial neural networks and adaptive neuro-fuzzy assessments for ground-coupled heat pump system. *Energy and Buildings*, *40*, 1074–1083. URL: `https://www.sciencedirect.com/science/article/pii/S0378778807002332`. doi:`https://doi.org/10.1016/j.enbuild.2007.10.002`.

Esen, H., Inalli, M., Sengur, A., & Esen, M. (2008b). Forecasting of a ground-coupled heat pump performance using neural networks with statistical data weighting pre-processing. *International Journal of Thermal Sciences*, *47*, 431–441. URL: `https://www.sciencedirect.com/science/article/pii/S1290072907000762`. doi:`https://doi.org/10.1016/j.ijthermalsci.2007.03.004`.

Esen, H., Inalli, M., Sengur, A., & Esen, M. (2008c). Performance prediction of a ground-coupled heat pump system using artificial neural networks. *Expert Systems with Applications*, *35*, 1940–1948. URL: `https://www.sciencedirect.com/science/article/pii/S0957417407004046`. doi:`https://doi.org/10.1016/j.eswa.2007.08.081`.

Esen, H., Ozgen, F., Esen, M., & Sengur, A. (2009). Artificial neural network and wavelet neural network approaches for modelling of a solar air heater. *Expert Systems with Applications*, *36*, 11240–11248. URL: `https://www.sciencedirect.com/science/article/pii/S0957417409002243`. doi:`https://doi.org/10.1016/j.eswa.2009.02.073`.

Fischer, T., & Krauss, C. (2018). Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, *270*, 654–669.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*, 1735–1780.

Hoffer, E., Hubara, I., & Soudry, D. (2017). Train longer, generalize better: closing the generalization gap in large batch training of neural networks. In *Advances in Neural Information Processing Systems* (pp. 1731–1741).

Hu, J., Shen, L., & Sun, G. (2018). Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 7132–7141).

Jang, H., & Lee, J. (2017). An empirical study on modeling and prediction of bitcoin prices with bayesian neural networks based on blockchain information. *Ieee Access*, *6*, 5427–5437.

Kalekar, P. S. et al. (2004). Time series forecasting using holt-winters exponential smoothing. *Kanwal Rekhi School of Information Technology*, *4329008*, 1–13.

Katsiampa, P. (2017). Volatility estimation for bitcoin: A comparison of garch models. *Economics Letters*, *158*, 3–6.

Kristjanpoller, W., & Minutolo, M. C. (2018). A hybrid volatility forecasting framework integrating garch, artificial neural network, technical analysis and principal components analysis. *Expert Systems with Applications*, *109*, 1 – 11. URL: `http://www.sciencedirect.com/science/article/pii/S095741741830294X`. doi:`https://doi.org/10.1016/j.eswa.2018.05.011`.

Lahmiri, S., & Bekiros, S. (2019). Cryptocurrency forecasting with deep learning chaotic neural networks. *Chaos, Solitons & Fractals*, *118*, 35–40.

Li, T. R., Chamrajnagar, A., Fong, X., Rizik, N., & Fu, F. (2019). Sentiment-based prediction of alternative cryptocurrency price fluctuations using gradient boosting tree model. *Frontiers in Physics*, *7*, 98.

Liang, Y., Ke, S., Zhang, J., Yi, X., & Zheng, Y. (2018). Geoman: Multi-level attention networks for geo-sensory time series prediction. In *IJCAI* (pp. 3428–3434).

Luo, Z., Cai, X., Tanaka, K., Takiguchi, T., Kinkyo, T., & Hamori, S. (2019). Can we forecast daily oil futures prices? experimental evidence from convolutional neural networks. *Journal of risk and financial management*, *12*, 9.

Matta, M., Lunesu, I., & Marchesi, M. (2015). Bitcoin spread prediction using social and web search media. In *UMAP workshops* (pp. 1–10).

McNally, S., Roche, J., & Caton, S. (2018). Predicting the price of bitcoin using machine learning. In *2018 26th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP)* (pp. 339–343). IEEE.

Minh, D. L., Sadeghi-Niaraki, A., Huy, H. D., Min, K., & Moon, H. (2018). Deep learning approach for short-term stock trends prediction based on two-stream gated recurrent unit network. *Ieee Access*, *6*, 55392–55404.

Mukhopadhyay, U., Skjellum, A., Hambolu, O., Oakley, J., Yu, L., & Brooks, R. (2016). A brief survey of cryptocurrency systems. In *2016 14th annual conference on privacy, security and trust (PST)* (pp. 745–752). IEEE.

Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *ICML*.

Nakamoto, S. (2019). *Bitcoin: A peer-to-peer electronic cash system*. Technical Report Manubot.

Patel, J., Shah, S., Thakkar, P., & Kotecha, K. (2015). Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques. *Expert Systems with Applications*, *42*, 259–268. URL: `https://www.sciencedirect.com/science/article/pii/S0957417414004473`. doi:`https://doi.org/10.1016/j.eswa.2014.07.040`.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Peng, Y., Albuquerque, P. H. M., Camboim de Sá, J. M., Padula, A. J. A., & Montenegro, M. R. (2018a). The best of two worlds: Forecasting high frequency volatility for cryptocurrencies and traditional currencies with support vector regression. *Expert Systems with Applications*, *97*, 177–192. URL: `https://www.sciencedirect.com/science/article/pii/S0957417417308163`. doi:`https://doi.org/10.1016/j.eswa.2017.12.004`.

Peng, Y., Albuquerque, P. H. M., de Sá, J. M. C., Padula, A. J. A., & Montenegro, M. R. (2018b). The best of two worlds: Forecasting high frequency volatility for cryptocurrencies and traditional currencies with support vector regression. *Expert Systems with Applications*, *97*, 177–192.

Radojičić, D., & Kredatus, S. (2020). The impact of stock market price fourier transform analysis on the gated recurrent unit classifier model. *Expert Systems with Applications*, *159*, 113565. URL: `http://www.sciencedirect.com/science/article/pii/S0957417420303894`. doi:`https://doi.org/10.1016/j.eswa.2020.113565`.

Rezaei, H., Faaljou, H., & Mansourfar, G. (2020). Stock price prediction using deep learning and frequency decomposition. *Expert Systems with Applications*, (p. 114332).

Sifat, I. M., Mohamad, A., & Shariff, M. S. B. M. (2019). Lead-lag relationship between bitcoin and ethereum: Evidence from hourly and daily data. *Research in International Business and Finance*, *50*, 306–321.

Sovbetov, Y. (2018). Factors influencing cryptocurrency prices: Evidence from bitcoin, ethereum, dash, litcoin, and monero. *Journal of Economics and Financial Analysis*, *2*, 1–27.

Tiwari, A. K., Kumar, S., & Pathak, R. (2019). Modelling the dynamics of bitcoin and litecoin: Garch versus stochastic volatility models. *Applied Economics*, *51*, 4073–4082.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems* (pp. 5998–6008).

Xu, Y., & Cohen, S. B. (2018). Stock movement prediction from tweets and historical prices. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 1970–1979).

Zhang, Z., Zhou, J., Song, Y., & Dai, H.-N. (2020). Cryptocurrencies price prediction using weighted memory multi-channels. In Z. Zheng, H.-N. Dai, X. Fu, & B. Chen (Eds.), *Blockchain and Trustworthy Systems* (pp. 502–516). Singapore: Springer Singapore.