Gold Price Forecast Based on LSTM-CNN Model

Conference Paper · August 2019

DOI: 10.1109/DASC/PiCom/CBDCom/CyberSciTech.2019.00188

4 authors, including:

Junhao Zhou
Macau University of Science and Technology
15 PUBLICATIONS **363** CITATIONS
SEE PROFILE

Hong-Ning Dai
Hong Kong Baptist University
296 PUBLICATIONS **18,497** CITATIONS
SEE PROFILE

Hao Wang
Norwegian University of Science and Technology, Gjøvik
199 PUBLICATIONS **4,920** CITATIONS
SEE PROFILE

Gold Price Forecast based on LSTM-CNN Model

Zhanhong He*, Junhao Zhou*, Hong-Ning Dai*, Hao Wang†
Macau University of Science and Technology, Macau SAR
calvinhzh@163.com; junhao_zhou@qq.com; hndai@ieee.org
†Norwegian University of Science and Technology, Gjøvik, Norway
hawa@ntnu.no

***Abstract*—An accurate prediction is certainly significant in financial data analysis. Investors have used a series of econometric techniques on pricing, stock selection and risk management but few of them have found great success due to the fact that most of them only are purely based on a single scheme. Recent advances in deep learning methods have also demonstrated the outstanding performance in the fields of image recognition and sentiment analysis. In this paper, we originally propose a novel gold price forecast method based on the integration of Long Short-Term Memory Neural Networks (LSTM) and Convolutional Neural Networks (CNN) with Attention Mechanism (denoted to LSTM-Attention-CNN model). Particularly, the LSTM-Attention-CNN model consists of three components: the LSTM component, Attention Mechanism and the CNN component. The LSTM component enables to harness the sequential order of daily gold price. Meanwhile, the Attention Mechanism assigns different attention weights on the new encoding method from LSTM component to enhance the extraction of the temporal and spatial features. In addition, the CNN component enables to capture the local patterns and abstract the spatial features. Extensive experiments on real dataset collected from World Gold Council show that our proposed approach outperforms other conventional financial forecast methods.**

I. Introduction

Finance is indispensable in our life and a precise forecast in financial area will facilitate the development of economy. Nowadays, economists or investigators will apply some mathematical methods to analyze the financial data with the aim of making prediction accurately. Naive Bayes algorithm is one of the traditional statistic tools to help the financial industries to predict what will be trendy financial products in the future. However, finance is one of the most computationally intensive fields in world. The Naive Bayes classifier can only deal with a small scale of data and influence the continuous features when it uses a binning procedure to process the data. Even though later many quantitative factor models, like Capital Asset Pricing Model (CAPM) or Fama-French three-factor model, have struggled since the financial crisis in 2008 and many traditional factors have ceased to be profitable. In addition, lots of practitioners are developing models, like linear regression model, that can dynamically learn from past data, dynamic models and ad-hoc factor-timing approaches, but they still face some valid criticisms (e.g., Asness (2016)) [1]. This is because, the financial domain is hugely complex and non-linear with plethora of factors influencing the estimation of the relationship between potential predictors and expected returns.

Deep learning has received extensive attention recently due to the superior performance especially in the fields of computer vision and natural language processing. The basic idea of deep learning is to use a set of data (refers to training data) to train a model and make prediction by using another set of data (refers to testing data). Although deep learning has demonstrated it success in the above mentioned fields, it is still questionable to apply deep learning in financial data analytics. For example, humans have no inborn ability to make prediction on finance or select a stock. Nevertheless, the core of deep learning is computing a set of input data, which may be economic data, accounting date or daily transaction record etc., mapping to a specific return. Hence, theoretically, no matter how irregular or non-linear the data we input, we can still find the relationships for the corresponding returns.

To this end, we propose and develop a deep learning model based on the combination model of Long Short-Term Memory Neural Networks (LSTM) and Convolutional Neural Networks (CNN) with Attention Mechanism to predict daily gold prices. The main contributions of this paper can be summarized as follows:

- We originally propose LSTM-CNN model with Attention Mechanism for daily gold price forecast.
- We conduct extensive experiments; the experimental results show that the proposed model outperforms other conventional models.
- We have found that LSTM-CNN model performs better than CNN-LSTM because it causes low feature loss than that of CNN-LSTM.
- We have also found that attention mechanism is a crucial part in our model, which can significantly improve the performance compared with pure LSTM-CNN model.

The remainder of this paper is organized as follows. In Section II, we describe the main approaches we adopt to do the daily gold price prediction task. Then the evaluation and results are discussed in Section III. Finally, we conclude our work and introduce the future research direction in Section IV.

II. Our Approach

Fig. 1 shows the proposed method used for prediction analysis of daily gold price. Firstly, the gold price per day can be obtained from World Gold Council and tabulated into Excel version. After being loaded into program and transformed as the array, a preprocessing step can be carried out to these datasets. Once the datasets have been prepared, all the pre-treated daily gold price data are then fed into Long Short-Term Memory Networks (LSTM) model to utilize sequences

of groups of daily gold price with specific width as input and generate a new series of encoding. At the same time, this new sequence of encoding will be sent to Attention Mechanism to enhance the temporal features. After that, Convolution Neural Network (CNN) model extracts spatial features from the output data and reduce the number of parameters or redundant features by passing them to Max Pooling layer. Lastly, the data will be sent to the last hidden layer will Fully-Connected Layer and Linear activation function, which will produce the eventual output that conducts the prediction of tendency for daily gold price.

A. Data Preprocessing

We conduct a preliminary analysis on the data of daily gold price. This dataset released by World Gold Council [2](WGC) contains 10471 daily gold price transaction record from Dec. 29, 1978 to Feb. 15, 2019 (only on trading day). Before feeding the input dataset to the next approach, we know that neural networks are sensitive to the diversity of input data. Thus, we exploit the Standard Scale method to standardize the data source. According to the document of scikit-learn [3], the standard scalar method will normalize the features so that each column/feature/variable will have mean = 0 and standard deviation = 1. The specific calculations of standard scalar are listed as following:

$$S(x_i) = \frac{x_i - \mu}{\sigma} \tag{1}$$

with mean:

$$\mu = \frac{1}{N}\sum_{i=1}^{N}(x_i) \tag{2}$$

and standard deviation:

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2} \tag{3}$$

where x_i stands for the value in gold price data over a period (e.g. a day).

As making an input data, we take the daily gold price data window length as α, whose dimension could be $\alpha \times 1$, into account. Specifically, in Fig. 2, we set α = 7 in our study, which means each input data set is consisted of 7 trading days. Based on the window length α, we transformed x_{t-1}, x_t, ..., x_{t+5} into S(x_{t-1}), S(x_t), ..., S(x_{t+5}). The corresponding values of output is S(x_{t-1}), which is also based on predict length. The detailed illustration Fig. 2 is shown as follows.

B. LSTM Component

The LSTM network is an artificial recurrent neural network (RNN), which is first proposed by Hochreiter & Schmidhuber [4] and improved by Alex Graves [5]. As shown in Fig. 1, the LSTM component (enclosed in green dash box) is a three-layer model (consisted of input layer, LSTM unit layer and output layer) with end-to-end feature. In this paper, we select β as the number of layers of LSTM and the number of layers β is

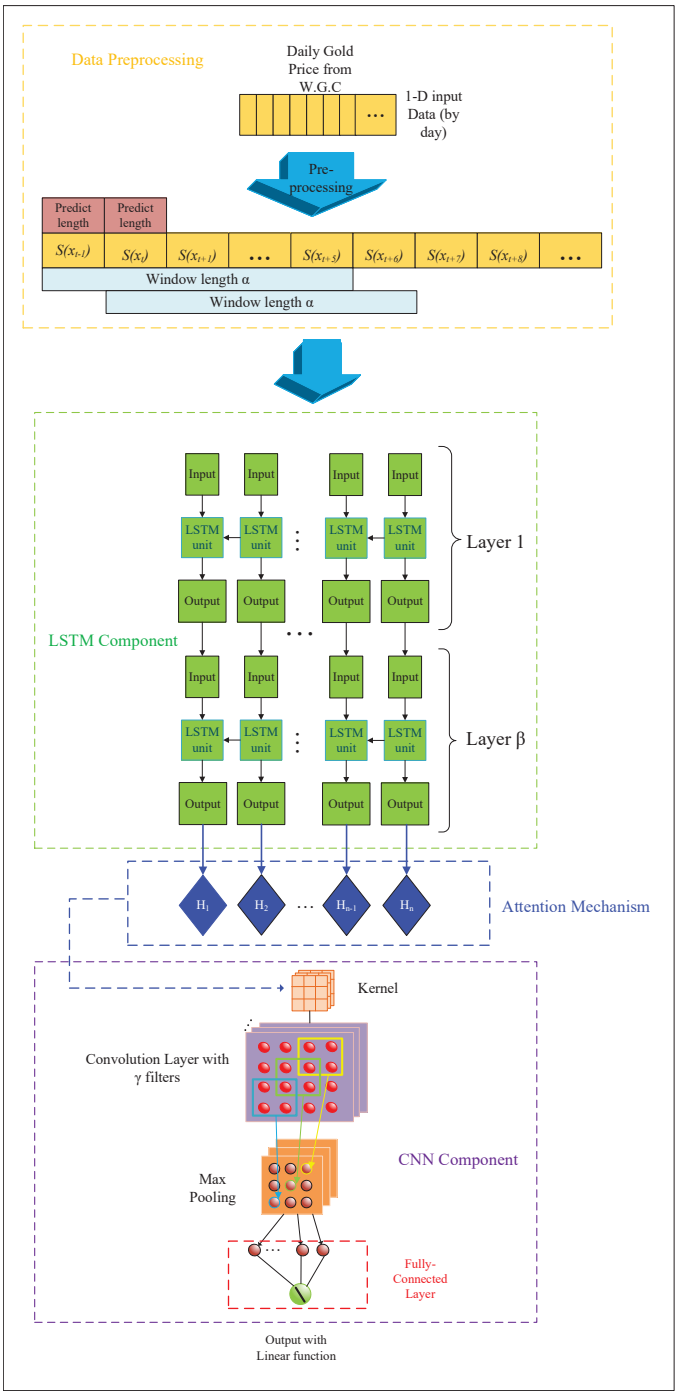

Fig. 1: LSTM-Attention-CNN framework

adjustable in the experiment. For time sequence T, the input sequence S(x_{T+1}), S(x_{T+2}), ⋯, S(x_{T+7}) wiil be entered into LSTM units to obtain a complete information from past time steps and future time steps by principle of LSTM. Basically, in Fig. 3, a common LSTM unit, which is enclosed in red dash box, is consisted of a **cell** and three gates (an **input gate**, an **output gate** and a **forget gate**) that control the flow from their memories. We can intuitively observe that the **cell** is responsible for keeping the tracks or we say "memories" of dependencies between the elements in the input. The role of

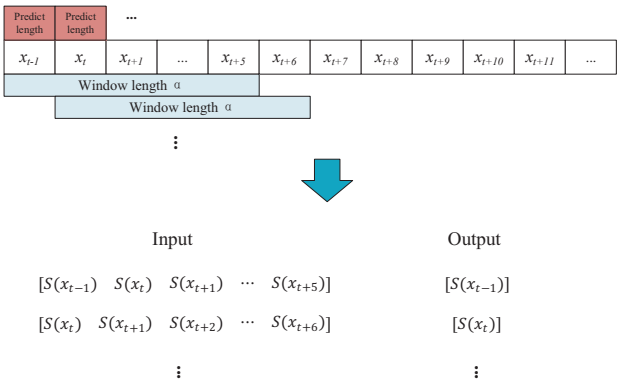

Fig. 2: Data Preprocessing

input gate (denoted as i_t) is to control the extent to which a new value flows into the **cell**. The **forget gate** (denoted as f_t) determines the extent to which values need to be remained or be forgotten (usually with a sigmoid function that produces 0 or 1). And the **output gate** (denoted as o_t) decides the extent to which the value in the cell is used to compute the output activation of the LSTM unit. The specific calculation is listed as following:

$$i_t = \sigma_g(W_i x_t + U_i h_{t-1} + b_i) \tag{4}$$

$$f_t = \sigma_g(W_f x_t + U_f h_{t-1} + b_f) \tag{5}$$

$$o_t = \sigma_g(W_o x_t + U_o h_{t-1} + b_o) \tag{6}$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \sigma_c(W_c x_t + U_c h_{t-1} + b_c) \tag{7}$$

$$h_t = o_t \circ \sigma_c(c_t) \tag{8}$$

where x_t is the input vector to the LSTM unit, h_t is the output vector of the LSTM unit. W_* , U_* and b_* are the weight matrices and bias vector parameters. Also, the initial values are c_0=0 and h_0=0 and the operator $\circ$ denotes the element-wise product. σ_g and σ_c refer to sigmoid function and hyperbolic tangent function respectively. At last the output layers integrate all the results as their output of LSTM component and send them to the next progress, Attention Mechanism.

C. Attention Mechanism

After being processed by LSTM component, we get a new sequence of encoding. But due to the large scale of time series input data, the more data we feed into LSTM component, the poorer effect will be resulted in the LSTM structure, which means the quality of memories that contain the processed information in each LSTM unit will decline. Therefore, it will directly affect that new sequence of encoding no matter on temporal or on spatial features. In order to better grasp the effective information from the new encoding and obtain significant temporal and spatial features, the Attention Mechanism will be introduced into the model. Attention Mechanism is a valid mechanism of distribution of probability weight. By assigning different attention probability weights, it pays more attention to some tendency or changes that happen in the training dataset of daily gold price and assigns more probability weights to enhance the features of new encoding. The mathematic calculation formulas of Attention Mechanism in this paper are shown as following:

$$e_{ij} = tanh(W_1 h_i + W_2 h_j + b_\alpha) \tag{9}$$

$$a_{ij} = softmax(e_{ij}) = \frac{exp(e_{ij})}{\sum_j exp(e_{ij})} \tag{10}$$

$$H_i = \sum_j a_{ij} h_j \tag{11}$$

where e_{ij} represents the relation between i^{th} value and j^{th} value. W is the weight, and b_α is bias parameter. a_{ij} denotes the attention weight of the i^{th} value versus j^{th} value by using the softmax function. H_i denotes the final state of the output after Attention Mechanism. We use the above formulas to calculate the corresponding weights of the new encoding and assign them to it so that we can generate another new encoding with enhanced features. After the Attention Mechanism processing, the output will be sent to the next step, CNN component.

D. CNN Component

As shown in Fig. 1, the CNN [6] component (enclosed in the purple dash box) is consisted of a convolution layer and a max pooling layer and it learns the global knowledge from the 1-D data. As proven by [7], [8] and [9], CNN has the advantages of extraction and reorganization. The spatial relationship of the training dataset of daily gold price is a vital characteristic for the prediction on the testing dataset of daily gold price. Thus, we apply CNN to extract the morphological features of data, which is the output from Attention Mechanism in our study. In this research, we only conduct one CNN layer on processing the data from Attention Mechanism. We will explain the structure of CNN component in details as following.

Single Convolution layer. One of the limitations of regular neural networks is the poor scalability due to the full-connectivity of neurons. CNN overcomes the disadvantages of regular neural networks by connecting each neuron to its neighboring neurons (P.S. not all the neurons). The parameters of convolutional layer usually consist of the filter (or kernels) numbers, filter length and a small set of neurons (also called as the receptive field) etc. During forwarding pass, each filter computes the dot product between the filter itself and the input dataset (1-D or 2-D metric). As a result, the network learns when it detects some specific types of feature, which are the characteristics, spatial position and weight sharing of the local view of input data. In this paper, we choose γ as the number of filters in convolutional layer and the number of filters γ is adjustable in the experiment. Moreover, we also design a unique kernel to fulfill the extracting function for daily gold price. We next describe the technical details as follows.

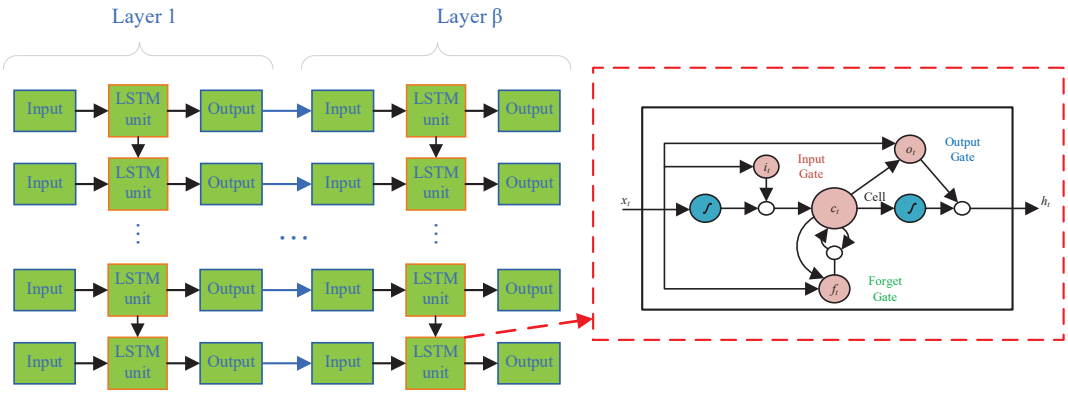

Fig. 3: LSTM detailed model

Max Pooling layer. The pooling layer has been typical used in CNN to reduce number of parameter (e.g. training weights and filters) and redundant features. Besides, a pool layer can be also used to control the convergence of neural networks (e.g. avoid overfitting). One of the most typical pooling operation is the *max pooling* [10]. In this way, the pooling layer will choose the maximum value of the field covered by the pooling filter. During the processed procedure, the pooling layer will route the weights to the corresponding filter with the highest value.

Fully-Connected layer. As shown in Fig. 1, the Fully-Connected layer, which is enclosed in the red dash box, calculates its own score by using the 1-D output data form max pooling layer according to the following equation:

$$y_j = \sum_{i=1}^{n} w_{i,j} x_i + b_1 \tag{12}$$

where y_j is the output of fully-connected layer in the j^{th} neuron, n is the length of 1-D input data (x), $w_{i,j}$ stands for the neuron weight between i^{th} input value and j^{th} neuron and b_1 is the bias. After the calculation, it will send these values to the connected units in the higher layer through an activation function to determine how much it contributes to the next step prediction. The activation function is given as follows:

$$u_j = g(y_j) = max(0, y_j) \tag{13}$$

where u_j is the output after activation function. In this paper, we use Rectified Linear Unit (ReLU) as the activation function, which will only activate the positive value. This function can effectively prevent the overfitting [11]. Before outputting the final results, the dataset will send to another activation function, called Linear activation. The activation function is given as following,

$$u_j = f(y_j) = u_j \tag{14}$$

this procedure is beneficial to sketching the line chart of prediction.

III. Experimental Results

A. Experiment Settings

1) Dataset description: In order to obtain a precise and authoritative dataset of gold price, we directly access to the website of World Gold Council, which is the authority on gold, to download the integrated gold price from Dec.29, 1978 to now, including yearly, quarterly, monthly and daily gold price in US dollar, EUR, RMB, HK dollar etc. per ounce. In particular, we only select the dataset specified in daily gold price (from Dec.29, 1978 to Feb.15, 2019, total 10471 trading days) in US dollar per ounce to make analysis prediction.

2) Model Setting: We set the window length of input to 7, which means each input matric contains 7 gold prices of trading days. The network weights are randomly initialized via using a truncated normal distribution (with mean μ = 0 and variance σ = 1.0). We then develop typical CNN model and LSTM model etc. All these models have the similar settings. Particularly, we fix the batch size as 80 and training epochs as 100. Furthermore, we choose the optimizer as "adam" for all the complier of Deep Learning models.

3) Experimental Performance Evaluation: In this paper, we will select three performance measures: the root mean square error (RMSE), the root mean absolute error (RMAE) and the mean absolute percentage error (MAPE) to evaluate the predictive power of our proposed models. RMSE is a useful performance metrics for revealing relatively large forecast errors [12]. RMAE is applied to measure the bias between practical and predictive model. MAPE is often to judge the performance of predictions in statistics. Meanwhile, we also set the loss of training set of each epoch to reflect on the performance of our proposed LSTM-Attention-CNN model influenced by different parameters. The loss is evaluated by MSE (mean square error).

In order to ensure the comparison fairness, we choose the same number of training times (epochs) for all Machine learning models and Deep learning models.

TABLE I: Performance Comparison with Other Conventional Schemes

Features	Training ratio = 60%			Training ratio = 80%		
	RMSE	RMAE	MAPE	RMSE	RMAE	MAPE
Support Vector Regression (SVR)	1.04E+03	3.18E+01	7.77E+01	1.04E+03	3.18E+01	7.77E+01
ARIMA	7.42E+02	2.51E+01	6.28E+01	8.84E+02	2.81E+01	5.81E+01
Deep Regression	2.64E+02	1.43E+01	1.58E+01	1.63E+02	1.16E+01	9.37E+00
CNN	5.03E+02	2.02E+01	3.24E+01	6.67E+01	7.77E+00	4.33E+00
LSTM	4.18E+02	1.78E+01	2.40E+01	6.04E+01	7.35E+00	3.23E+00
CNN-LSTM	6.29E+02	2.21E+01	3.76E+01	1.08E+02	8.60E+00	4.75E+00
LSTM-CNN	4.17E+02	1.76E+01	2.33E+01	4.25E+01	5.52E+00	2.12E+00
LSTM-Attention-CNN	4.18E+02	1.77E+01	2.34E+01	**3.07E+01**	**4.66E+00**	**1.54E+00**

B. Model Comparison

Baseline models: We choose Deep Regression, Support Vector Regression (SVR), Autoregressive Integrated Moving Average model (ARIMA), Convolutional Neural Network (CNN), Long Short-Term Memory Neural Networks (LSTM) and two combinational models* for comparison.

***CNN-LSTM:** This scheme is consisted of an initial CNN layer and then a latter LSTM layer. Its processing order is the converse version of our approach without Attention Mechanism.

***LSTM-CNN:** This scheme is consisted of an initial LSTM layer and then a latter CNN layer. Its processing order is the version of our approach without Attention Mechanism.

Table I presents the performance of our LSTM-Attention-CNN model and other methods. We conduct three groups of experiments with the training ratio of 60% and 80% respectively. It is worth to mention that our training ratio is defined by the ratio of the size of training sample to that of all the daily golden price in time series. We choose parameters α = 7, β = 1, γ = 64 for our LSTM-Attention-CNN model. For each group of training ratio, we evaluate 8 models by 3 performance metrics, including RMSE, RMAE and MAPE.

From Table I, we can learn that, in the training ratio of 60%, our proposed LSTM-Attention-CNN model has better scores than most of compared models like SVR, ARIMA, CNN and LSTM etc. in corresponding to RMSE, RMAE and MAPE. When the training ratio is 80%, our proposed method overperforms than any other models. For instance, the LSTM-Attention-CNN model can achieve the minimum RMSE value with 3.07E+01 compared with other schemes when the training ratio is 80%. This implies that our proposed method has the better performance of daily gold price prediction than that of other conventional schemes, no matter in financial models or typical machine learning models.

In addition, when independently comparing LSTM-Attention-CNN model with LSTM-CNN model, the scores of our proposed method in RMSE, RMAE and MAPE are all less than that of LSTM-CNN model respectively in these three evaluation metrics. This implies that Attention Mechanism can effectively improve the outcome from LSTM component.

For better visualization on what the above performance metrics results reflect, we also sketch the prediction line charts

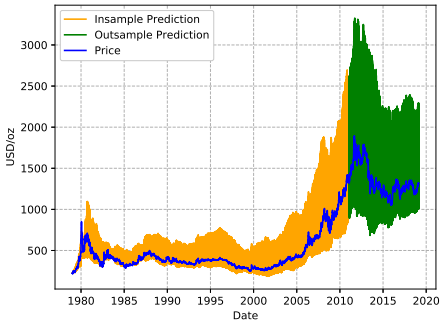

Fig. 4: Prediction of ARIMA with training data in 80%

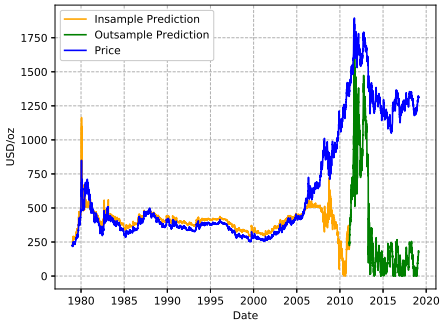

Fig. 5: Prediction of SVR with training data in 80%

based on the training ratio of 80% and 100 epochs (except ARIMA and SVR) for the corresponding models. Note that, in each line chart, "In-sample Prediction", which is in orange, refers to the data that we use to train or fit the model; "Out-sample Prediction", which is in green, refers to the data that we use to valid or test the model; "Price", which is in blue, illustrates the real price of gold in USD/oz.

When comparing LSTM-CNN model with CNN-LSTM model, the performance of LSTM-CNN model is always better than the performance of CNN-LSTM no matter in training ratio, in 3 performance metrics or in the illustration of prediction line chart. These results seem to indicate that the score difference between these two models is not coincident and the

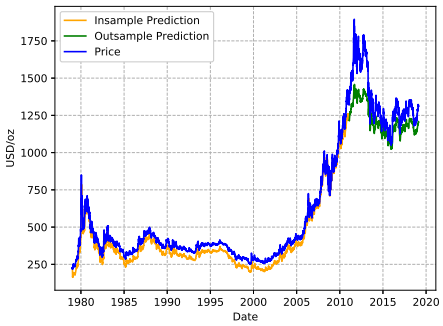

Fig. 6: Prediction of Deep Regression with training data in 80%

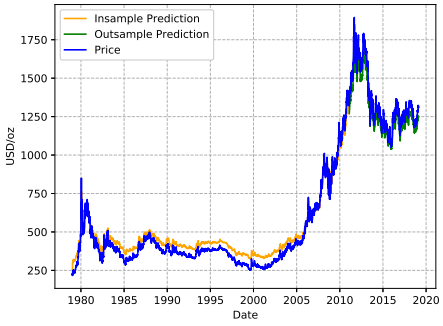

Fig. 7: Prediction of CNN with training data in 80%

ordering of the layer will play a crucial role on how well they perform.

The LSTM-CNN model has well-performance since the initial layer, LSTM, plays the role as the encoder that it records the sequence of daily gold price as long as it has already processed. In other words, for every input unit there will be an output unit that has the memory of not only the original information, but also of the information of all other processed data record. Then the CNN layer will extract the local features by making use of the optimization of the original input data

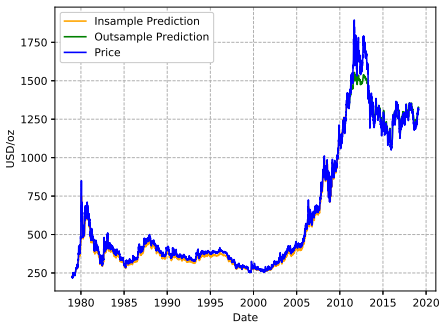

Fig. 8: Prediction of LSTM with training data in 80%

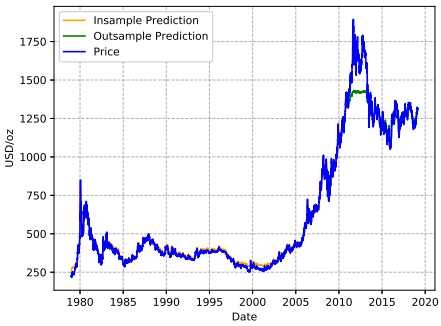

Fig. 9: Prediction of CNN-LSTM with training data in 80%

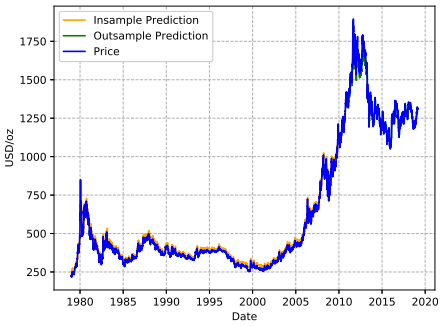

Fig. 10: Prediction of LSTM-CNN with training data in 80%

so as to improve the prediction performance.

On the contrary, the first layer of CNN-LSTM model, the convolutional layer, will disorganize the sequence of our daily gold price for a certain extent when extracting the typical patterns. Hence, if a set of time series data is disordered and fed to the LSTM layer, it is meaningless for this approach since the LSTM cannot give play to its advantage but just act as another fully-connected layer. As we can observe from the Table I, the performance of CNN-LSTM model is even worse than a single model of LSTM.

C. Parameter study

For model itself, the change of the structure may have immediate impact on its performance. To study this, we then investigate the impacts of various parameters on the performance of our proposed LSTM-Attention-CNN scheme.

1) Effect of α: α is a parameter controlling the size of window length for each input metric. To investigate the impact of α on the prediction results, we set the size of window length to 7, 10 and 13 respectively. At the same time, we fix β = 1 and γ = 64. We conduct the experiment with the training ratio with 80% and we visualize the loss of every epoch (from 0 to 1200) for better comparison.

In view of external factor, the variation of window length of input may cause different experimental results. Fig. 12 shows

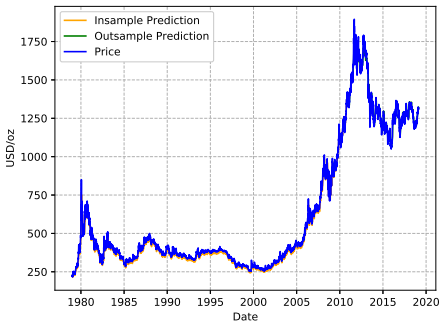

Fig. 11: Prediction of LSTM-Attention-CNN with training data in 80%

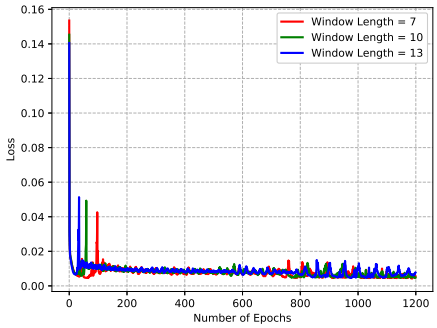

Fig. 12: The loss of LSTM-Attention-CNN Model with different window length of input size

us that the loss of training set declines dramatically as the number of epochs ranges from 0 to 50. When comparing three different window length of input, we can learn that window length equals to 7 (in red) has a slightly better performance than the other two window lengths. Since when the epochs = 50, the loss line in red reaches to its bottom at about 0.005. As the number of epochs grows, the loss increases again and stays steady at almost 0.01. Fig. 12 indicates that even through enlarging the size of input and providing more data, the network may not get better performance on learning the features and making prediction. Also, we should control the number of training epochs at a specific range to meet the best condition of the model.

2) Effect of β: β is a parameter controlling the number of LSTM layers in LSTM component. To investigate the impact of β on the prediction results, we set the number of LSTM layers to 1 and 2 respectively. At the same time, we fix α = 7 and γ = 64. We conduct the experiment with the training ratio at 80% and we visualize the loss of every epoch (from 0 to 1200) for the better comparison.

Theoretically, with the growth of the number of layers in neural network, the model could get better performance since stacked layers may be beneficial to feature extraction. However, after we add one more LSTM layers, Fig. 13 does not show the result that we expect. The loss of 1 LSTM-Attention-CNN model is similar to that of 2 LSTMs-Attention-CNN model within 25 epochs, while the green one reaches its bottom but the red one still descends slightly and meets the smallest value at about 0.005. Ranging the number of epochs from 100 to 1200, 1 LSTM-Attention-CNN model performs steadily at the loss of about 0.01, whereas the loss of 2 LSTMs-Attention-CNN model fluctuates between 0.02 and 0.025. This implies that one LSTM layer is suitable for our model. If we add one more layer of LSTM, it may result in over-extraction.

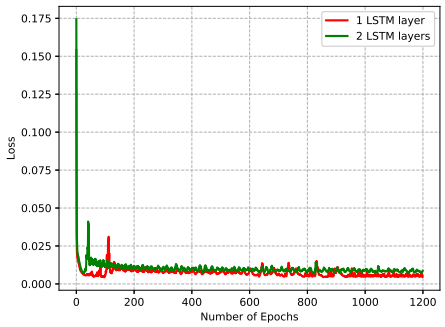

Fig. 13: The loss of LSTM-Attention-CNN Model with different number of LSTM layers

3) Effect of γ: γ is a parameter controlling the number of filters in CNN component. To investigate the impact of γ on the prediction results, we set the number of filters in CNN to 32, 64 and 128 respectively. At the same time, we fix α = 7 and β = 1. We conduct the experiment with the training ratio with 80% and we visualize the loss of every epoch (from 0 to 1200) for the better comparison.

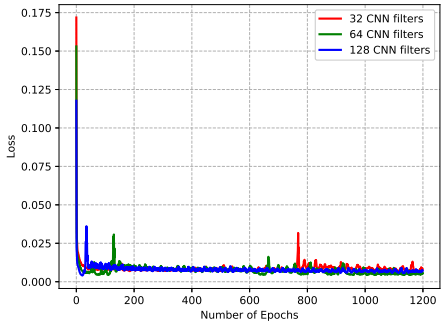

Fig. 14: The loss of LSTM-Attention-CNN Model with different number of CNN filters

From Fig. 14, it is obvious that the model with more CNN filters has better performance. We can learn that the loss values of the model with 128 CNN filters are lower and achieve faster convergence compared with the other two models between 0 and 200 epochs. However, as the number of epochs increases, the loss of model with 128 CNN filters fluctuates sharply in the latter 100 epochs and keeps steady for the rest of epochs, which is higher than that of the model with 64 CNN filters. The result supports that the larger the hidden size, the richer

information can be processed from the model and performs better, but it should be within appropriate epochs or it may lead to counteractive at the performance of the model.

IV. Conclusion

In this paper, we propose an integration of LSTM and CNN neural networks with Attention Mechanism to predict the tendency of daily gold price. In particular, our LSTM-Attention-CNN model consists of a LSTM component, Attention Mechanism and CNN component. The LSTM component can generate the new encoding and *memorize* the previous processed data, and Attention Mechanism can strengthen the weights of each part in that encoding, while the CNN component has the advantage of extracting the local features. We conduct extensive experiments on realistic daily gold price data to evaluate the performance of the proposed model. The results show that our proposed model outperforms conventional methods such as ARIMA, deep regression, SVR, CNN. Furthermore, it would be beneficial to test different types of RNNs aside from LSTM component for our model. For instance, using bidirectional LSTM networks on LSTM-Attention-CNN model might obtain a better result.

References

[1] K. Rasekhschaffe and R. Jones, "Machine learning for stock selection," *Financial Analysts Journal, Forthcoming*, 2019.

[2] https://www.gold.org.

[3] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html.

[4] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735

[5] A. Graves, A. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, May 2013, pp. 6645–6649.

[6] Y. Lecun and Y. Bengio, *Convolutional networks for images, speech, and time-series*. MIT Press, 1995.

[7] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: an overview and application in radiology," *Insights into Imaging*, vol. 9, no. 4, pp. 611–629, Aug 2018. [Online]. Available: https://doi.org/10.1007/s13244-018-0639-9

[8] N. Tajbakhsh, J. Y. Shin, S. R. Gurudu, R. T. Hurst, C. B. Kendall, M. B. Gotway, and J. Liang, "Convolutional neural networks for medical image analysis: Full training or fine tuning?" *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1299–1312, May 2016.

[9] D. Jaswal and K. P. Soman, "Image classification using convolutional neural networks," 2014.

[10] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *J. Mach. Learn. Res.*, vol. 12, pp. 2493–2537, Nov. 2011.

[11] Z. Wang, W. Yan, and T. Oates, "Time series classification from scratch with deep neural networks: A strong baseline," 2016.

[12] P. Tsang, P. Kwok, S.-O. Choy, R. Kwan, S. Ng, J. Mak, J. Tsang, K. Koong, and T.-L. Wong, "Design and implementation of nn5 for hong kong stock price forecasting," *Engineering Applications of Artificial Intelligence*, vol. 20, pp. 453–461, 06 2007.

[13] S. Lathuilière, P. Mesejo, X. Alameda-Pineda, and R. Horaud, "A comprehensive analysis of deep regression," 2018.

[14] K.-j. Kim, "Financial time series forecasting using support vector machines," *Neurocomputing*, vol. 55, pp. 307–319, 09 2003.

[15] S. M. Madge, "Predicting stock price direction using support vector machines," 2015.

[16] K. Yunus, T. Thiringer, and P. Chen, "Arima-based frequency-decomposed modeling of wind speed time series," *IEEE Transactions on Power Systems*, vol. 31, no. 4, pp. 2546–2556, July 2016.

[17] A. Vaccaro, T. H. M. EL-Fouly, C. A. Cañizares, and K. Bhattacharya, "Local learning-arima adaptive hybrid architecture for hourly electricity price forecasting," in *2015 IEEE Eindhoven PowerTech*, June 2015, pp. 1–6.

[18] T. Fischer and C. Krauss, "Deep learning with long short-term memory networks for financial market predictions," *European Journal of Operational Research*, vol. 270, 12 2017.

[19] D. Soutner and L. Müller, "Application of lstm neural networks in language modelling," in *TSD*, 2013.