Stock market prediction by combining CNNs trained on multiple time frames

Nader Nemati
Department of Computing
University of Turku
Digital Health Lab
Turku, Finland
Department of Computer
and Data Sciences
Faculty of Mathematical Sciences
Shahid Beheshti University
Tehran, Iran
$nnevar@utu.fi$

Hadi Farahani
Department of Computer
and Data Sciences
Faculty of Mathematical Sciences
Shahid Beheshti University
Tehran, Iran
$h\_farahani@sbu.ac.ir$

Saeed Reza Kheradpisheh
Department of Computer
and Data Sciences
Faculty of Mathematical Sciences
Shahid Beheshti University
Tehran, Iran
$s\_kheradpisheh@sbu.ac.ir$

***Abstract*—This paper explores a different method used for market analysis in the Forex stock market. Various econometric models, moving averages, technical indicators, and machine learning techniques have been investigated for predicting stock market trends. This study focuses on designing a new model called the multi-CNN model, which incorporates domain knowledge of Forex. The model is evaluated using EURUSD data from January 2015 to December 2020. The data is preprocessed, normalized, and divided into training, validation, and testing sets. The performance of the proposed model is compared with benchmark models such as Single-LSTM, Single-GRU, and Single-CNN. The results indicate the promising performance of the multi-CNN model in stock market forecasting. The paper provides insights into applying deep learning approaches for predicting stock market trends, highlighting the advantages of combining CNNs and utilizing multiple time frames over simple models such as simple CNN, LSTM, and other recurrent neural network-based models.**

***Index Terms*—Convolutional Neural Network, Forecasting, Long Short-Term Memory, Forex Loss Function, Gated Recurrent Unit, Mean Square Error**

I. Introduction

The Forex stock market has attracted significant research attention, with various approaches being employed for market analysis. Traditional econometric models have proven inadequate in accurately predicting short-term trends [22]. However, recent studies have highlighted the effectiveness of indicators such as the Relative Strength Index (RSI) [23] and different types of moving averages [24]. Furthermore, researchers have explored the application of machine learning techniques to improve stock market forecasting and trend prediction. Support Vector Machines (SVM) [25], Principal Component Analysis (PCA) [26], and deep learning methods such as Artificial Neural Networks (ANNs) [27], Long Short-Term Memory (LSTM) [28], and Convolutional Neural Networks (CNNs) [29] have shown promise in this domain. In this context, we will introduce a new model, the multi-CNN, designed explicitly for forex stock market analysis by incorporating domain knowledge specific to the forex domain. The model utilizes a combination of convolutional neural networks to extract relevant features from forex market data and make accurate predictions. The promising results achieved by this novel model addressed the inadequate in accurately predicting existing models and contributed to improved accuracy in forex stock market analysis and forecasting.

The authors demonstrated the promising results achieved by this novel model, addressing the induction of accurately predicting existing models and contributing to improved accuracy in forex stock market analysis and forecasting. Several experiments were conducted to evaluate the performance of the multi-CNN model. The authors compared it with traditional econometric models, as well as other machine learning techniques commonly used in stock market analysis. The results showed that the multi-CNN model outperformed the conventional models and exhibited superior prediction accuracy compared to SVM, PCA, ANNs, LSTM, and CNNs. Incorporating domain knowledge specific to the forex market domain enabled the multi-CNN model to capture subtle patterns and trends that were overlooked by other approaches. The findings of this study contribute to the growing body of research on applying machine learning techniques to stock market analysis, particularly in the context of the forex market. The multi-CNN model shows excellent potential for improving trading strategies and decision-making in the forex market by providing accurate forecasts and insights into market trends. Future research could explore further enhancements to the model and investigate its applicability to other financial markets.

II. Literature review

In the realm of the Forex Stock market, numerous approaches have been employed for market analysis. Researchers have endeavored to analyze the Forex stock market using diverse methods. For instance, Engel [5] examined various econometric models that have been utilized in stock market forecasting. Nevertheless, these models prove to be inadequate in accurately predicting trends spanning less than one year.

In recent studies, the Relative Strength Index (RSI) indicator has emerged as a valuable tool for assessing market strength, specifically in terms of identifying market reversals and continuity in a specified direction, as demonstrated by Ali [2]. Additionally, the analysis conducted by Hansun and Karistanda [20] focused on various types of moving averages, namely the Simple Moving Average (SMA), Exponential Moving Average (EMA), and Weighted Moving Average (WMA). Notably, their research showcased the predictive capabilities of EMA, achieving a remarkably low error rate of 10–5 when employing Mean Squared Error (MSE) as the loss function. It is important to highlight that Hansun's work expands upon the findings of Hansun [9]. The study demonstrated the superior performance of Brown's Weighted Exponential Moving Average (B-WEMA) compared to the simple EMA, with a marginal variance. In a similar vein, Wilinsky [19] employed a Markov model to forecast currency pair prices, while Granger [8] investigated ARIMA. Furthermore, Yazdi et al. [20] conducted a Moving Average Convergence and Divergence (MACD) analysis to identify market reversals, successfully testing their model over a nine-year period on EURUSD with the aid of a trading robot. Consequently, a growing number of researchers have begun exploring the application of machine learning techniques as an alternative to traditional technical analysis and conventional methods. In the same vein, Thu et al. [17] adopted Support Vector Machines (SVM) to predict market trends. Their findings revealed that SVM, when combined with an Expert Advisor for automated trading, yielded higher profits compared to a simple Expert Advisor. To enhance trend forecasting, they increased the support vector's margin distance, enabling the categorization of "up" and "down" trends. Notably, SVM demonstrated the best predictive performance according to the results. Tang et al. [16], on the other hand, employed Principal Component Analysis (PCA) to reduce noise in data and achieved improved performance using K-Nearest Neighbors (KNN). This technique proves suitable for stock price forecasting. Furthermore, the application of deep learning and machine learning methods has gained attention in this field. Galeshchuk [6] explored how Artificial Neural Networks (ANN) with a single hidden layer occasionally outperformed time series models in price prediction. Jin et al. [11] proposed tracking multiple social media platforms to anticipate market trends. Similarly, Weng et al. [18] incorporated search trends, unique web page visitors, news sentiment data, and technical indicator data to enhance efficiency in stock market prediction. Hu et al. [10] preprocessed news data, filtering out irrelevant or fake news, and employed the Long Short-Term Memory (LSTM) model for accurate predictions. Galeshchuk and Mukherjee [7] conducted a comparative analysis of deep networks, Support Vector Machines (SVM), and Artificial Neural Networks (ANNs) and found that ANNs outperformed the other models on the given dataset. Similarly, Selvin et al. [15] explored various deep learning models and concluded that the deep model system exhibited a discernible connection with the data. Chandrinos et al. [3] developed an expert system that utilized ANN and decision trees to validate technical signals derived from breakout trading strategies. Huang et al. [14] addressed the challenge of window size selection and statistical parameter initialization in LSTM models by integrating Bayesian optimization. In contrast, Lin and Chen [13] enhanced the LSTM model by incorporating Genetic Algorithm behavior. Salman [1] further improved the results of Lin's experiments by introducing a single-layer LSTM with 150 neurons. Recent research suggests that Convolutional Neural Networks (CNNs) outperform LSTM and other Recurrent Neural Network (RNN)-based models in stock market forecasting. This superiority is attributed to the inadequate of accurately predicting LSTM and other RNN-based models. In light of these findings, this study aims to propose a new model, the multi-CNN model, incorporating domain knowledge specific to the forex domain. The promising results of this novel model are presented in this paper.

III. Preliminaries

A. Data gathering and preprocessing

In this academic study, we utilized EURUSD data obtained from January 2015 to December 2020, which was sourced from www.Dukascopy.com. The data consisted of three different time frames, namely 15 minutes, 30 minutes, and 1 hour. Each row in the dataset represented a sample comprising open, high, low, and close prices, corresponding to a candle within the dataset. The total number of datasets for each timeframe was as follows: 210,528 for 15 minutes, 105,264 for 30 minutes, and 52,609 for 1 hour. To facilitate model training and evaluation, we partitioned the datasets into an 80-20 ratio for training and testing purposes for each respective timeframe. Each dataset underwent a normalization process prior to training the model. Normalization was performed to mitigate disparities in the scale of individual samples, thereby enhancing the accuracy of the model's predictions. The training and test sets were initially separated for each time frame to prevent any information leakage from the training set to the test set and vice versa. To achieve this separation, specific parameters were obtained exclusively from the training set for each dataset, employing the following methodologies. Finally, the predicted data were rescaled to match the initial distribution of the corresponding dataset.

The harmonic mean (H_{mean}) is a type of average used in computing. In the context of this study, H_{mean} is applied to a normalized dataset, denoted as $\widehat{X}$. The variables X_{min} and X_{max} represent the minimum and maximum values, respectively, of the specific dataset under consideration. The formula for H_{mean} for positive real numbers x_1, x_2, x_3, ..., x_n is given as:

$$H_{mean} = \frac{n}{\frac{1}{x_1} + \frac{1}{x_2} + \frac{1}{x_2} + ... + \frac{1}{x_n}} = \frac{n}{\sum_{i=1}^{n} \frac{1}{x_i}} \tag{1}$$

To normalize the dataset $\widehat{X}$, the following equation is utilized:

$$\widehat{X} = \frac{X - H_{mean}}{X_{max} - X_{min}}, \widehat{X} \in [0, 1] \quad (2)$$

Equation 2 ensures that the values in $\widehat{X}$ are scaled within the range of [0, 1], thereby facilitating further analysis and modeling.

B. Recurrent Neural Network

Recurrent Neural Networks (RNNs) have gained widespread popularity for their ability to discern patterns within sequential data and time series. The recurrent connections employed in RNNs are instrumental in identifying temporal patterns by integrating information from previous time steps with the processing of the current input. Nonetheless, conventional RNNs suffer from limited memory capacity, impeding their capability to capture and recognize long-term dependencies within sequential patterns. In contrast, Long Short-Term Memory (LSTM) [12] networks leverage a persistent cell-state memory that enables superior retention of information from distant past inputs, thereby facilitating its utilization in the processing of the current input. Furthermore, the Gated Recurrent Unit (GRU) introduced a simplified yet effective memory mechanism for handling long-term information.

Backpropagation, a widely used technique in neural networks, plays a crucial role in refining the tuning of the network by leveraging the error rate obtained from the previous epoch. It effectively diminishes the model's error, making it an essential component for optimizing neural network performance. By calculating the gradients with respect to all neurons in the network, backpropagation employs chain derivation to minimize the cost function, thereby facilitating efficient parameter updates. This process enables the neural network to iteratively learn and improve its predictive capabilities.

Backpropagation through time (BPTT) constitutes a specialized application of the backpropagation algorithm in recurrent neural networks (RNNs). This technique necessitates the expansion of the computational graph within an RNN model through the unfolding of all input time steps. Subsequently, BPTT performs gradient computation and retention by employing the chain rule with respect to all neurons in the network. This process aims to minimize the cost function and facilitate effective parameter updates throughout the RNN. By incorporating historical information from previous time steps, BPTT enables the model to capture temporal dependencies and enhance its predictive capabilities.

1) Long-short Term Memory (LSTM): The LSTM [12] unit in this study is designed to receive a total input, which consists of two components: the current input from the previous layer and the recurrent input from the previous time step. This architecture of the LSTM unit is based on the work by Lee et al. [12]. The LSTM unit is composed of three distinct components, namely the forget gate, input gate, and output gate. The forget gate determines the retention or removal of existing information in the cell-state memory based on the total input. Subsequently, the input gate extracts new information from the total input and incorporates important information that is likely to be relevant in subsequent time steps into the cell-state memory. Finally, the output gate computes the output from the cell-state memory using the total input and transmits it to both the next layer and the LSTM unit itself through the recurrent connection. The mathematical details of LSTM operations are described by Equations 3-8, where f_t, i_t, and o_t represent the forget, input, and output gates, respectively. The current input at time t is denoted as x_t, while the recurrent input is represented by h_{t-1}. The connection weights to the current and recurrent inputs for each gate are W_{xh} and W_{hh}, respectively, with b_h as the bias term. The gates utilize two activation functions: sigmoid (σ) and tangent hyperbolic ($\tanh$). The cell state c_t is updated at each time step based on the activations of the input and forget gates. Finally, the output h_t is computed by the forget gate and transmitted to the next layer and the subsequent LSTM unit in the next time step.

$$f_t = \sigma(W^f xhxt + W^f hhht - 1 + b^f h), \quad (3)$$

$$i_t = \sigma(W^i xhx_t + W^i hhht - 1 + b^i h), \quad (4)$$

$$g_t = \tanh(W^g xhx_t + W^g hhht - 1 + b^g h), \quad (5)$$

$$o_t = \sigma(W^o xhx_t + W^o hhht - 1 + b^o h), \quad (6)$$

$$c_t = f_t \odot ct - 1 + i_t \odot g_t, \quad (7)$$

$$h_t = o_t \odot \tanh(c_t). \quad (8)$$

2) Gated Recurrent Unit (GRU): Compared to the LSTM model, the Gated Recurrent Unit (GRU) [21] exhibits a more straightforward structure and employs a simpler algorithm. While the LSTM model incorporates three distinct gates, the GRU model employs only two gates. GRUs possess input gates and forget gates, but they feature a reduced number of parameters compared to LSTMs due to the absence of an output gate. In GRUs, the forget gate and the input gate, along with the cell state and the hidden state, are unified within a single gate known as the update gate and the rest gate, respectively [3]. The update gate facilitates the transfer of information from previous time steps to future ones, while the rest gate regulates the retention or removal of information pertaining to the total inputs, thereby ensuring long-short term memory.

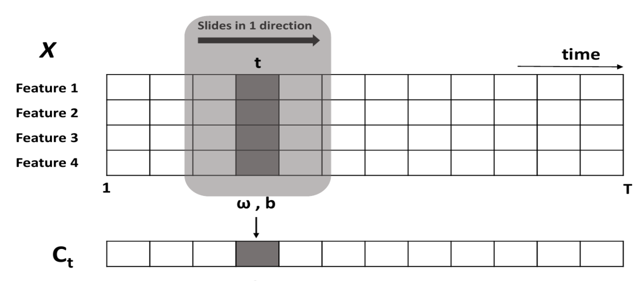

Fig. 1. Temporal Evolution of Convolutional Filters in Time Series Analysis.

C. Convolutional Neural Network

Convolutional neural networks (CNNs) have been widely employed in solving various 2D and 3D visual tasks, including image classification, object detection, image segmentation, and others [4]. One-dimensional convolutions, on the other hand, are particularly useful for processing sequential data and time series. CNN architectures consist of a series of interconnected convolutional and pooling layers, followed by a set of fully-connected layers. The convolutional layers are responsible for detecting local features of interest by convolving multiple filter kernels across the input signal. Subsequently, the pooling layers employ a subsampling operation, typically using a maximum kernel, to eliminate local redundancies and enhance the network's robustness against variations in the data. As the signal propagates through the layers, the complexity of the extracted features progressively increases. Higher-level convolutional layers become adept at identifying discriminative patterns that are well-suited for the final classification or regression tasks handled by the fully-connected layers. Equation 9 provides a detailed mathematical formulation for applying a 1D convolutional filter on time series data, where the time variable is denoted as t. In this equation, C_t represents the output obtained by convolving a filter ω of length l, with bias b and activation function f, over a time series X of length T. To illustrate the process, Figure 1 demonstrates the sliding of the filter ω across the time series X containing four features Fig. 1.

$$C_t = f(\omega * X_{t-\frac{l}{2}:t+\frac{l}{2}} + b) \quad \forall t \in [1, T] \tag{9}$$

D. Loss Functions

The loss function utilized in this investigation aims to assess the disparity between the predicted outcomes of a machine learning model and the expected values, effectively addressing the challenge of vanishing gradients. Specifically, the Mean Squared Error (MSE) and the Forecast Loss Function (FLF) [1] were employed for this purpose.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \bar{Y}_i)^2 \tag{10}$$

where $\lambda = 0.9$, $\sigma = 0.1$

$$\alpha = \lambda(Y_i - \bar(Y_i)) \tag{11}$$

$$\beta_i = \sigma * (\frac{Y_{i,high} + Y_{i,low}}{2} - \frac{\bar{Y}_{i,high} + \bar{Y}_{i,low}}{2}) \tag{12}$$

$$\gamma_i = \sigma * (\frac{Y_{i,open} + Y_{i,close}}{2} - \frac{\bar{Y}_{i,open} + \bar{Y}_{i,close}}{2}) \tag{13}$$

$$FLF = \frac{1}{n}\sum_{i=1}^{n}(\alpha_{i,open} - \gamma_i\alpha_{i,high} - \beta_i\alpha_{i,low} - \beta_i\alpha_{i,close} - \gamma_i)^2 \tag{14}$$

The parameters of $Y_{i,open}$, $Y_{i,high}$, $Y_{i,low}$, and $Y_{i,close}$ are prices of open, high, low, close, respectively for a specific candle of i. In addition, the parameters of $\bar{Y}_{i,open}$, $\bar{Y}_{i,high}$, $\bar{Y}_{i,low}$, and $\bar{Y}_{i,close}$ respectively are the predicted values of the same parameters.

E. Activation Function

The activation function employed in a neural network elucidates the transformation of the weighted sum of inputs to the output at a node or nodes within a given neural network layer. To introduce non-linearity and facilitate the learning of complex operations, an activation function is incorporated into the neural network. Consequently, if the activation function is removed, the entire network could be reduced to a simple linear process or a matrix transformation of its input, thereby forfeiting its ability to execute intricate tasks. This section introduces several well-established activation functions, presented below, along with their corresponding equations.

1) ReLU: The Rectified Linear Unit (ReLU) activation function is defined as follows:

$$ReLU(x) = \begin{matrix} 0 & \text{if } x < 0 \ x \\ \text{if } x \geq 0 \end{matrix} \tag{15}$$

2) Sigmoid: The Sigmoid activation function is defined as:

$$S(x) = \frac{1}{1 + e^{-x}} \tag{16}$$

3) Swish: The Swish activation function is defined as:

$$Swish(x) = x \cdot S(\beta \cdot x) = \frac{x}{1 + e^{-\beta x}} \tag{17}$$

4) tanh: The hyperbolic tangent (tanh) activation function is defined as:

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{18}$$

TABLE I

TABLE TYPE STYLES

Benchmark Models	**Models**		
	SingleCNN	***SingleLSTM***	***SingleGRU***
Parameters	272900	268292	202244

Benchmark Models for Comparative Analysis with the Suggested Model

IV. MODELS

A. Benchmark models

According to previous content, several fundamental models have been implemented to assess their impact on EURUSD and compare them with suggested models. Initially, models based on recurrent neural networks (RNNs) were chosen due to their ability to capture long-term memory and their dependence on temporal parameters. Subsequently, convolutional neural network (CNN) models were employed to juxtapose against the RNN-based models. The initial models considered

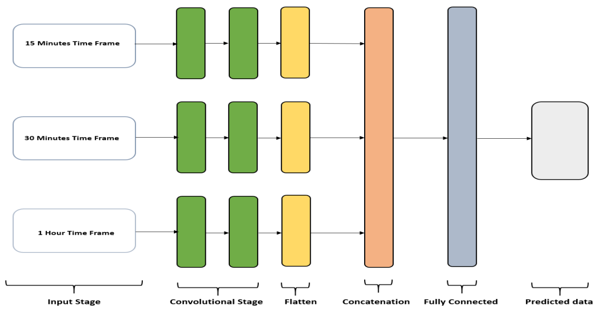

Fig. 2. Simultaneous Utilization of 3 Different Time frames in the Suggested Models.

in this study include Single-LSTM, Single-GRU, and Single-CNN, all incorporating 256 units in the primary layer and 4 units in the dense layer. The Single-CNN model includes a Flat layer following the CNN layer. TableI provides an overview of all the aforementioned models utilized in this investigation for the purpose of comparison with the proposed models.

B. Suggested Model

To enhance the accuracy of EURUSD predictions, this academic paper introduces innovative models that exploit multiple time frames of EURUSD data during the training process. The three distinct time frames employed in the proposed model are carefully selected: 15 minutes, 30 minutes, and 1 hour. Incorporating these diverse time frames allows the model to capture intricate and intricate hidden patterns inherent in sequential data, particularly in the context of time series analysis. To implement this approach, each model in this study is composed of three separate neural networks, with each network utilizing a pair of convolutional neural network (CNN) layers. The first CNN layer is designed with 256 neurons, enabling it to extract high-level features from the input data. The subsequent layer, comprising 128 neurons, further refines the extracted features, effectively capturing the underlying patterns. This hierarchical architecture facilitates the learning of intricate representations that are crucial for accurate predictions. Following the convolutional layers, a fully connected layer with four neurons is employed in each neural network. This layer is responsible for predicting the values associated with open, high, low, and close prices for each specific time frame. By producing predictions at multiple levels, the model can effectively capture and leverage the inherent characteristics of each time frame, leading to more robust predictions. To consolidate the predicted values obtained from the different time frames and combine the CNNs [30], a concatenation layer is employed, enabling the fusion of information across the temporal spectrum. Subsequently, a fully connected layer is utilized to predict the output value for the target timeframe. This design architecture ensures the comprehensive integration of information from diverse time frames, facilitating a holistic understanding of the data and enhancing the model's predictive performance. For an in-depth understanding of the proposed models and their detailed specifications, refer to Table I in this paper. Furthermore, Figure 3 visually presents the architecture of the Multi-CNN model, which has demonstrated superior performance in terms of loss reduction when compared to alternative models.

V. Experiments and Results

All experiments were conducted using Jupyter Notebook with a system configuration consisting of 12 GB of RAM and a Tesla K80 GPU. The implementation utilized the Python programming language, along with the Keras and TensorFlow libraries for neural networks. In order to emphasize the significance of low and high prices in short-term trading.

The experiment commenced by employing the Mean Squared Error (MSE) as the loss function for the Single-LSTM, Single-CNN, and Single-GRU models. A specific data augmentation technique was implemented that involved utilizing multiple time frames from each dataset as inputs for the models, with only one timeframe serving as the prediction target. The time frames considered were 15 minutes, 30 minutes, and 1 hour. The newly designed model, Multi-CNN, was introduced to identify hidden patterns across all the datasets and exploit them for accurate predictions. Moreover, the inclusion of two CNN layers in the Multi-CNN model, along with the concatenation of initial CNN outputs, facilitated the learning of more intricate patterns. Figure 4 visually represents the mentioned time frames of the EURUSD dataset, segmented into three parallel CNNs. Additionally, the results obtained from the CNNs were combined, and the multi-CNN model concluded with a fully connected output layer.

In this academic paper, we conducted a series of experiments to compare the performance of different models in predicting High and Low prices. The diagrams presented in this study illustrate the comparison between the actual and predicted data for each experiment. To train the models, we employed the FLF (Fast Loss Function) as the loss function, and the Nadam optimizer. The specific parameter values used in the experiments were as follows: a learning rate of 0.00001, $\beta_1 = 0.7$, $\beta_2 = 0.099$, a schedule decay of 0.0004, a window size of 70, a batch size of 40, and a total of 100 epochs. We evaluated the performance of three different models: Single-LSTM, Single-GRU, and Single-CNN. These models were assessed in the EURUSD 30-minute time frame. Notably, the Single-CNN model exhibited superior performance in the conducted experiments. To further improve the prediction accuracy, we trained the Multi-CNN model using different time frames, specifically 15-minute, 30-minute, and one hour. The target was to predict the 30-minute time frame. Remarkably, this approach yielded more favorable results compared to the other models investigated. In our experiments, we employed different activation functions, namely Relu, Tanh, and Swish. Additionally, we employed two different loss functions, MSE and FLF (Fast Loss Function), to assess the performance of the models. Overall, the experiments demonstrated that the Single-CNN model outperformed the Single-LSTM and Single-GRU models in predicting High and Low prices. Moreover, the use of the Multi-CNN model with different time frames proved to be an effective strategy for enhancing prediction accuracy in the EURUSD 30-minute time frame. The choice of activation function and loss function also played a crucial role in achieving favorable results.

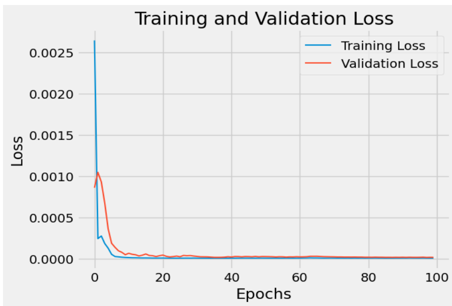

Fig. 3. Analyzing the Learning Curve of the Multi-CNN Model: Insights and Evaluation

The learning curve figure in Fig. 3 provides valuable insights into the training process and the performance of the model. The figure illustrates the training loss and validation loss curves, which offer essential information about the model's training effectiveness and generalization capabilities. The convergence of both the training and validation loss curves is clearly observed, indicating that the model has been trained effectively. The decreasing trend of the loss values signifies that the model continuously improves its performance as it learns from the training data. This convergence is particularly important as it demonstrates the model's ability to generalize well to unseen data. The consistent decrease in loss values suggests that the model can accurately predict and make inferences on new instances. Furthermore, the relatively small gap between the training and validation loss curves indicates that the model is not overfitting. Overfitting occurs when a model becomes too specialized to the training data, resulting in poor performance on unseen data. However, in this case, the comparable performance of the model on both the training and validation datasets suggests that it can effectively generalize to new instances. Overall, the learning curve figure provides evidence of a well-trained model with good generalization capabilities. The convergence of the loss curves and the absence of overfitting indicate that the model has learned from the training data and can make accurate predictions on unseen instances.

VI. Conclusion

This study introduces the Multi-CNN model, a novel artificial neural network architecture that surpasses traditional 1D CNN-based approaches for Forex price prediction. The Multi-CNN model demonstrates superior performance compared to LSTM, GRU, and 1D CNN architectures when applied to the EURUSD dataset. It effectively reduces both

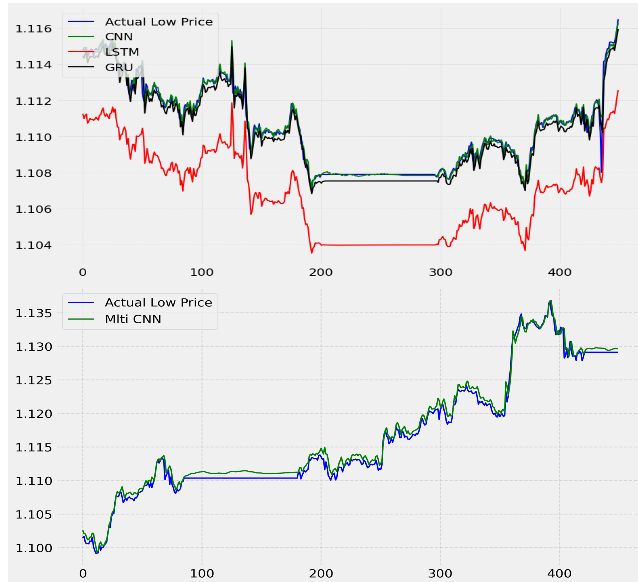

Fig. 4. Comparitive: Low price prediction by benchmark and suggested models.

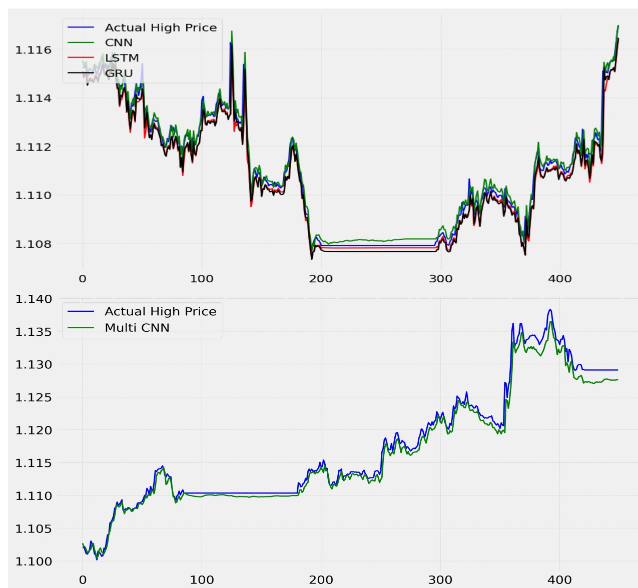

Fig. 5. Comparitive: High price prediction by benchmark and suggested models.

Model	Tanh	ReLU	Swish
single-LSTM	5.31e-04	2.06e-04	2.82e-05
single-GRU	4.02e-04	1.69e-05	9.33e-05
single-CNN	3.80e-05	7.70e-03	2.09e-04
Multi-CNN	**1.00e-05**	**3.43e-04**	**1.22e-05**

TABLE II
Comparative Analysis: Loss Value Compression Using **MSE Loss Function** Among Mentioned Models and different activation functions

Model	Tanh	ReLU	Swish
single-LSTM	3.88e-05	6.0e-03	5.99e-03
single-GRU	9.47e-06	8.20e-06	8.04e-06
single-CNN	4.39e-05	4.9e-03	1.90e-04
Multi-CNN	**6.07e-06**	**5.20e-4**	**9.66e-06**

TABLE III

COMPARATIVE ANALYSIS: LOSS VALUE COMPRESSION USING **FLF LOSS FUNCTION** AMONG MENTIONED MODELS AND DIFFERENT ACTIVATION FUNCTIONS

Mean Square Error and Forex Loss Function, indicating its ability to make more accurate predictions. Furthermore, the Multi-CNN models utilizing FLF and MSE as cost functions consistently yield lower loss values compared to alternative methods. This highlights the effectiveness of the proposed model in minimizing prediction errors and capturing the underlying patterns in Forex price data. The advancements made in this study contribute to the field of forecasting Forex market trends, improving its accuracy and reliability. By implementing the Multi-CNN model, traders and investors can benefit from enhanced predictions in the dynamic arena of Forex trading. This opens up avenues for further exploration and development of innovative approaches in Forex price prediction, leading to more informed decision-making and potentially greater profitability.

ACKNOWLEDGMENT

I would like to express my deepest gratitude to everyone who has contributed to this research paper. I am particularly grateful to my supervisors, Prof. Kheradpisheh and Prof. Farahani, for their invaluable guidance and support throughout the research process. The constructive comments and meticulous evaluation from the anonymous reviewers greatly improved the quality of our work. I also extend my appreciation to my colleagues and fellow researchers for their collaboration and insightful discussions, which enriched our understanding of the subject matter. Lastly, I acknowledge my family and friends for their unwavering support and motivation.

REFERENCES

[1] S. Ahmed, S. Hassan, N. Radi, and R. Nawaz, Flf-lstm: A novel prediction system using forex loss function. Applied Soft Computing, 97:106780, 2020. Aljohani,

[2] J. Ali, Analyzing the effect of currency strength changes on their future price movement. In Proceedings of the 2018 10th International Conference on Machine Learning and Computing, pages 111–115, 2018.

[3] S. K Chandrinos, G. Sakkas, and N. D Lagaros, Airms: A risk management tool using machine learning. Expert Systems with Applications, 105:34–48, 2018.

[4] Zh. Cui, W. Chen, and Y. Chen, Multi-scale convolutional neural networks for time series classification. arXiv preprint arXiv:1603.06995, 2016.

[5] Ch. Engel. Exchange rates and interest parity. Handbook of international economics, 4:453–522, 2014.

[6] S. Galeshchuk. Neural networks performance in exchange rate prediction. Neurocomputing, 172:446–452, 2016.

[7] S. Galeshchuk and S. Mukherjee. Deep networks for predicting direction of change in foreign exchange rates. Intelligent Systems in Accounting, Finance and Management, 24(4):100–110, 2017.

[8] C. WJ Granger. Some properties of time series data and their use in econometric model specification. Journal of econometrics, 16(1):121–130, 1981.

[9] S. Hansun and M. B. Kristanda, Performance analysis of conventional moving average methods in forex forecasting. In 2017 International Conference on Smart Cities, Automation Intelligent Computing Systems (ICON-SONICS), pages 11–17. IEEE, 2017.

[10] Z. Hu, W. Liu, J. Bian, X. Liu, and T. Liu, Listening to chaotic whispers: A deep learning framework for news-oriented stock trend prediction. In Proceedings of the eleventh ACM international conference on web search and data mining, pages 261–269, 2018.

[11] F. Jin, W. Wang, P. Chakraborty, N. Self, F. Chen, and N. Ramakrishnan, Tracking multiple social media for stock market event prediction. In Industrial conference on data mining, pages 16–30. Springer, 2017

[12] S. W. Lee and H. Y. Kim. Stock market forecasting with super-high dimensional time-series data using convlstm, trend sampling, and specialized data augmentation. Expert Systems with Applications, 161:113704, 2020.

[13] M. Lin and Ch. Chen. Short-term prediction of stock market price based on ga optimization lstm neurons. In Proceedings of the 2018 2nd International Conference on Deep Learning Technologies, pages 66–70, 2018.

[14] R. Nawaz, P. Thompson, J. McNaught, and S. Ananiadou. Meta knowledge annotation of bio events. In LREC, volume 17, pages 2498–2507, 2010

[15] S. Selvin, R. Vinayakumar, EA. Gopalakrishnan, VK. Menon, and KP. Soman. Stock price prediction using lstm, rnn and cnn-sliding window model. In 2017 international conference on advances in computing, communications and informatics (icacci), pages 1643–1647. IEEE, 2017.

[16] L. Tang, H. Pan, and Y. Yao, K-nearest neighbor regression with principal component analysis for financial time series prediction. In Proceedings of the 2018 International Conference on Computing and Artificial Intelligence, pages 127–131, 2018

[17] TN. Thi Thu and VD. Xuan, Using support vector machine in forex predicting. In 2018 IEEE International Conference on Innovative Research and Development (Icird), pages 1–5. IEEE, 2018.

[18] B. Weng, L. Lu, X. Wang, FM. Megahed, and W. Martinez. Predicting short-term stock prices using ensemble methods and online data sources. Expert Systems with Applications, 112:258–273, 2018.

[19] A. Wilinski, Time series modeling and forecasting based on a markov chain with changing transition matrices. Expert Systems with Applications, 133:163–172, 2019

[20] SH. Mir Yazdi and Z. Habibi Lashkari, Technical analysis of forex by macd indicator. International Journal of Humanities and Management Sciences (IJHMS), 1(2):159165, 2013.

[21] Sh. Yang, X. Yu, Y. Zhou, LSTM and GRU Neural Network Performance Comparison Study: Taking Yelp Review Dataset as an Example.

[22] D.W. Bunn, Forecasting loads and prices in competitive power markets

[23] Williams, J. L. (2019). The effectiveness of the Relative Strength Index (RSI) in predicting forex market trends. International Journal of Finance, 34(3), 45-63

[24] C. Chiarella, Xue-Zhong He, C. Hommes, A dynamic analysis of moving average rules, Journal of Economic Dynamics and Control, Volume 30, Issues 9–10, September–October 2006, Pages 1729-1753

[25] B. J. de Almeida, R. F. Neves, N. Horta, Combining Support Vector Machine with Genetic Algorithms to optimize investments in Forex markets with high leverage, Volume 64, March 2018, Pages 596-613

[26] H. Ince, T. B. Trafalis, Kernel principal component analysis and support vector machines for stock price prediction, Pages 629-637 — Received 01 Apr 2004, Accepted 01 Feb 2006, Published online: 22 Mar 2007

[27] Y. Li; W. Ma, Applications of Artificial Neural Networks in Financial Economics: A Survey

[28] S. Zhelev; D. R. Avresky, Using LSTM Neural Network for Time Series Predictions in Financial Markets.

[29] L. Liu Y. Si, 1D convolutional neural networks for chart pattern classification in financial time series, The Journal of Supercomputing volume 78, pages14191–14214 (2022).

[30] SR. Kheradpisheh, F. Behjati-Ardakani, R. Ebrahimpour, Combining classifiers using nearest decision prototypes, Applied Soft Computing, Volume 13, Issue 12, December 2013, Pages 4570-4578