CS9-1 AI-GUIDED FINANCIAL TRADING

Final Report

Information Technology Capstone Project

COMP5703/5707/5708

Group Members

1. Shunfei (Aurelius) Zhang (510330529)
2. Jialin (Chris) Li (510564892)
3. Dezheng Fang (480330688)
4. Zhijun Xing (500083662)
5. Bingtao Zhang (490041116)

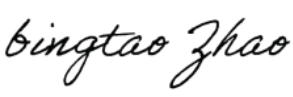
CONTRIBUTION STATEMENT

Our group, taking project CS9-1 AI-GUIDED FINANCIAL TRADING, with group members Shunfei Zhang, Jialin Li, Dezheng Fang, Zhijun Xing and Bingtao Zhang, would like to state the contributions each group member has made for this project during semester 1 2021:

- Shunfei Zhang: Development on GAN model and associated back-testing with Dezheng Fang
- Dezheng Fang: Development on GAN model and associated back-testing with Shunfei zhang
- Jialin Li: Development on Transformer and TCN model and associated back-testing with Zhijun Xing
- Zhijun Xing: Development on Transformer and TCN model and associated back-testing with Jialin Li
- Bingtao Zhao: Development on CNN+Bi-LSTM+Attention model and associated back-testing

All group members agreed on the contributions listed on this statement by each group member.

Signatures:



ABSTRACT

Advancements in modern computing have paved the way for applications of deep learning in a wide number of industries, particularly within finance. Stock exchanges such as the foreign exchange (forex) have garnered particularly high focus due to their volatility and potentially high stakes investments. There is motivation then, to investigate further into the application of deep learning methodologies to make accurate predictions on the forex market. This study utilises historic price data of four major currency pairs as features along with technical analysis indicators. This proposal provides analyses of various deep learning methods and their suitability to solving the market price prediction problem. Models using combinations of Long Short-Term Memory (LSTM), Convolutional Neural Networks (CNN), Transformer, Temporal Convolutional Network (TCN) and Generative Adversarial Networks (GANs) are proposed based on this justification. Based on those models, we also implemented some existing ideas and novel ideas to improve the model. This report details the processes of research, selection, development and evaluation of the deep learning models with a proposed timeline for project completion.

TABLE OF CONTENTS

1. INTRODUCTION

The application of deep learning models to make predictions on the stock market is by no means a novel concept. The rapid advancements of computing systems and the internet have given way to sophisticated methods that are able to produce faster, more accurate predictions and in high frequency (Zhou et al., 2018). Such models typically use historic price data to train and find hidden patterns in stock movement to predict future market direction to an extent. For instance, while there are plenty of models that generate significant results in a theoretical sense, these models are usually not nearly as successful when applied in a real-world trading context. This is largely due to the high volatility of markets, whose fluctuating stock prices are influenced by a myriad of variables including industry trends, macroeconomic factors, company news and performance, and unexpected events such as natural disasters (Hu, Zhao & Khushi, 2021).

The foreign exchange (forex) market is one of the largest markets in the world by trading volume. Compared with the stock market, Forex operates on a more frequent basis, and it is a lot more volatile. Investors have been putting effort to find out the underlying patterns of the exchange rate and wanted to profit from it. However, the nature of such a fluctuating market has made this process so difficult, and may therefore more sophisticated methods are required (Hu, Zhao & Khushi, 2021). .

The motivation for this project is to thus develop a deep learning model that can be profitable when tested in a simulated real-time forex trading environment. The development of a good model can provide guidelines to the investors. The predictions generated can help all investors control the investment risks and maximise the profitability.

This report firstly reviewed the literature which is frequently used in the industry and then proposed some ideas based on the existing models.

2. RELATED LITERATURE

2.1 Technical Indicators

Technical indicators are the signals produced by the heuristic open, high, low or close price of the currencies. These pattern-based signals will be helpful for the traders to better analyse the trend or decide the proper entry and exit point into the market. These signals can also provide the specialists with necessary information to predict the future price movement of the market.

As Technical indicators are always quantitative analysis, it is natural to incorporate them in the financial prediction models or AI trading systems.

There are Four basic types of technical indicators: They are trending indicators, momentum indicators, volume indicators and volatility indicators. The Trending indicators attempt to identify the direction of previous and current trends, and also give the alert about the potential changes in trend. The Momentum indicators want to show the overbought and oversold levels. The volume and volatility indicators were trying to show the abnormal volume and volatility.

As there exists hundreds of indicators, it is necessary to be selective when involving them into the models.

MACD (Moving Average Convergence & Divergence)

MACD usually uses the closing price of the short-term period (12-day EMA) and long-term period(26-day EMA) to indicate the change in the bullish and bearish trend of the market and the convergence and separation status to study the timing of buying and selling signals.

Talib API:

dif, dem, histogram = MACD(close, fastperiod=12, slowperiod=26, signalperiod=9)

Equation:

$$dif\ (MACD)\ =\ EMA(12\ timesteps)\ -EMA(26\ timesteps)$$
$$dem\ (Signal)\ =\ EMA(MACD, 9\ timesteps)$$

$$Histogram = dif - dem$$

RSI (Relative Strength Index)

RSI analyzes the intention and strength of market buying and selling orders by comparing the average closing gains and average closing losses in a certain period, reflecting whether the market goes bullish or bearish. It also considers the normal distribution, and the value range should be higher than 30 and lower than 70. If the number exceeds 70, it shows the overbought signal; the market price may naturally adjust. But note that the RSI function has an irregular period.

Talib API:

real = RSI(close, timeperiod=14)

Equation:

$$RSI = 100 - \frac{100}{1 + RS}$$

$$RS = \frac{Average\ Gain\ in\ N\ timesteps}{Average\ Loss\ in\ N\ timesteps}$$

ADX (Average Directional Movement Index)

ADX indicator judges the consolidation, oscillation and unilateral trends. It needs to calculate the DM(Directional Movement) first. DM represents that today's stock price is greater than the maximum of yesterday's stock price volatility. If the price goes higher, it is recorded as +DM; if it is a downtrend, it is recorded as -DM. Otherwise, DM=0.

ADX can't tell the direction of the trend, but if used in conjunction with other indicators and the trend exists, ADX can confirm the existence of the trend and measure its strength.

Talib API:

real = ADX(high, low, close, timeperiod=14)

Equation:

+DI14 = +DM/TR14*100

-DI14 = +DM/TR14*100

DX = |(+DI14)-(-DI14)| / |(+DI14)+(-DI14)|

$$+DI = 100 * \frac{Current\ High - Previous\ High}{Average\ True\ Range}$$

$$-DI = 100 * \frac{Previous\ Low - Current\ Low}{Average\ True\ Range}$$

$$DX = 100 * \frac{|+DI - (-DI)|}{|+DI + (-DI)|}$$

$$ADX = \frac{Prior\ ADX * 13 - Current\ ADX}{14}$$

MA (Moving Average)

Moving average is the sum of closing prices for a certain period of the time divided by the period and reduces the influence of insignificant fluctuations. For example, the daily MA10 refers to the closing price within 10 days and divided by 10. Moving average only responds to the change after the events happened, it cannot be used for a prediction.

Talib API:

real = MA(close, timeperiod=30, matype=0)

EMA & SMA (Exponential Moving Average & Simple Moving Average)

EMA is a trend indicator; the older data points are multiplied by a smaller multiplier, so EMA responds more quickly to recent changes. It is used to judge the changing trend of the price trend in the future.

In SMA, the data of each time period has the same weight. Once the latest data points are successively added, the oldest data points will be deleted

Talib API:

real = EMA(close, timeperiod=30)

real = SMA(close, timeperiod=30)

Equation:

$$EMA = (Value_{Current} * (\frac{Smoothing}{1 + Days})) + EMA_{last} * (1 - (\frac{Smoothing}{1 + Days}))$$

The most common smoothing factor is 2.

BBANDS (Bollinger Bands)

Bollinger Bands is named after the surname of the founder of the indicator(John Brin). It is a medium and long-term technical indicator for judging the price of movement trend of trading products. It uses statistical principles to find the standard deviation of currency prices and their confidence intervals, thus determining the range of stock price fluctuation and future trends. Also, it will use the wavebands to show the safe high and low prices, so it is called Bollinger bands. The width of this band varies with the magnitude of stock price fluctuations, and the band will become wider or narrow due to the stock price changes.

Usually, when the price moves above the middle rail line, it indicates a long trend, and when the prices move below the rail line, it indicates that there is a short trend.

Talib API:
upperband, middleband, lowerband = BBANDS(close, timeperiod=5, nbdevup=2, nbdevdn=2, matype=0)
Equation:

$$TP = (High + Low + Close) / 3$$

$$Upper\ BAND = Moving\ Average(TP, 20) + 2 * \sigma(TP, 20)$$

$$Lower\ BAND = Moving\ Average(TP, 20) - 2 * \sigma(TP, 20)$$

CCI (Commodity Channel Index)

CCI indicator precisely measures whether the stock price had exceeded the normal distribution range. When it was in the range from +100 to -100, it had little significance.

When the CCI indicator breaks through the +100 lines from top to bottom and re-enters the normal range, it shows that the rising phase of market prices may be over, and it might enter a period of turbulence and consolidation. When the CCI indicator curve breaks through the -100 line and enters into the abnormal range(which is always the

oversold zone), it indicates that the weak state of the market price has been formed and suggests holding short orders and waiting for higher profits.

Talib API:

real = CCI(high, low, close, timeperiod=14)

Equation:

$$TP = \sum_{i=1}^{P} ((High + Low + Close)/3)$$

$$CCI = \frac{TP - MA}{0.015 * (\sum_{i=1}^{P} |TP - MA|)/P}$$

2.2 Literature Review

Machine learning technologies have been applied to the financial trading domain for a long time. The autoregressive integrated moving average (ARIMA) model is one of the classical machine learning applications which achieve rather good results in financial prediction (Box and Jenkins, 1976). Also, more advanced machine learning models have been raised in decades. For example, the SVM has been applied to predict the trend of the stock market on a weekly level (Wen et al., 2005). ANN can also be used with the help of ARIMA to get better results with ARIMA to deal with the linear pattern and ANN to deal with the non-linear part of the model. (Areekul et al., 2010).

Transformer was first proposed by Vaswaniet et al. (2017). First applied to the field of natural language processing, the model proposed to use only attention modules instead of traditional RNN models. Wu et al. (2020) applied Transformer to time series forecasting. Their study showed that Transformer outperformed other popular time series models such as ANN and LSTM.

Bai et al. (2018) proposed a Temporal Convolutional Network (TCN) to be applied to time series prediction. Traditional convolutional neural networks are generally considered less suitable for modelling time-series problems, mainly due to the limitation of their convolutional kernel size, which does not capture long-time dependent information well. They reconsider the general connection between sequence modelling and recurrent networks by proposing temporal domain convolution, and they compare it with a variety of RNN structures and find that TCN can meet or even exceed RNN models on a variety of tasks.

Kim, T. and H. Y. Kim (2019) indicates that using a combination of SC-CNN and ST-LSTM to obtain the time series data and image characteristics together to get the better prediction of time series data. The SC-CNN model obtains the critical feature from the different stock images by implementing the residual learning and bottleneck architecture. The ST-LSTM model tries to find the optimal model by analysing the combination of the closing price and daily trading volume.

Wang, J., et al. (2018) indicate the using of discrete Fourier transform and Z-transform, to make the time series data into the form of frequency spectrum and use the mWDN (multilevel Wavelet Decomposition Network) to decompose the time series from high to low, and that enabled the computer to capture the frequency factors for deep learning. But this Multilevel Wavelet method has used some of the future information; thus, this method could not be used in the experiment.

In the article produced by Lin, Z. et al. (2020), they introduced the concept of DoppelGANger. The fidelity of the general GAN model is poor, and it performs poorly on the long sequence's dataset, which was heavy-tailed with variable length. The DoppelGANger can solve this problem by generating high-quality synthetic time series data; it generates the small batches of growing sequences using a combination of MLP and recursive networks to capture the time dependence and decouple the normalization. So it can automatically normalize the feature within the given range and then add those features to the sample dataset.

3. RESEARCH/PROJECT PROBLEMS

3.1 Research/Project Aims & Objectives

This project aims to predict the next close price of foreign exchange rate using advanced Deep Learning models. The project would also implement some novel ideas on our baseline models. Together with our own trading strategy we aim to achieve the highest profitability.

We have four base models in this project. As our predictions reveal the overall trend of the market, buyers will be able to choose the trend in the time span that they wish to trade. In this way, buyers are able to go long in the uptrend and short sell in the downtrend. The objective is to make the most profit. The project aims to achieve a

smaller Root Mean Squared Error (RMSE) during training and out-of-sample testing, and to generate higher profitability during back-testing.

3.2 Research/Project Questions

The main problem faced by stakeholders is the need for a model that can produce accurate predictions on the next close price and hence leading to maximum profitability. We have considered which technical indicators to include as features as well as the kinds of methodologies to use for model evaluation. We also implemented some novel ideas on the four base models.

This research aims to answer the question:

- How can we achieve an out-of-sample RMSE lower than 0.0005
- How can we achieve on average 10 pips profit per day in back-testing

3.3 Research/Project Scope

This research problem is in the scope of next close price predictions based on historical forex data and technical analysis indicators only. This project makes use of four major currency pairs to make predictions. This data will be processed and input through the proposed models to be analysed and evaluated based on the methodologies described in the next section.

4. METHODOLOGIES

4.1 Modelling Methodology

4.1.1 GAN

Adversarial generative network was first proposed by Goodfellow in 2014 [15]. GAN is inspired by the Two-player Zero sum Game in game theory. Usually, the adversarial generative network consists of a generator G and a discriminator D. Among them, the function of generator G is to learn the distribution of input data and map a section of input features into its output, in this task, the prediction of future price. The function of discriminator D is to distinguish whether the input price is the real future price or the future price that the generator generates. The optimization process of the network is adversarial. The training goal of generators is to generate more realistic price forecasts, and the training goal of discriminators is to try to determine whether prices

are true or false. The training process adopts an adversarial mechanism and trains generator and discriminator alternately. The discriminator parameters will be fixed when training the generator's parameters. Instead, generator parameters are fixed when training discriminator parameters. The training goal of both generators and discriminators is to maximize the error of the other side. As a result, the generator will produce a more realistic future price, and the discriminator will be better able to judge. When D's discriminant ability is improved to a certain extent and the data source cannot be correctly identified, generator G can be considered to have learned the real data distribution.

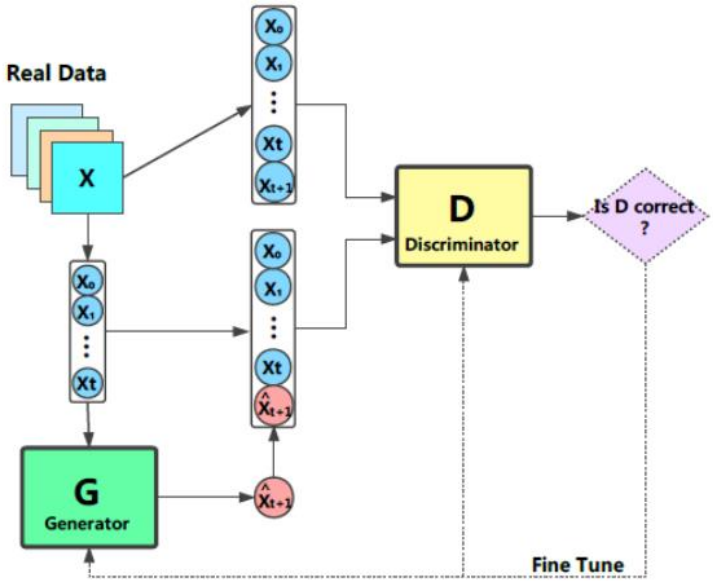


Fig. 1. GAN architecture (Zhang et al., 2019)

Generator:

In this task, the input data is forex data, so the GRU is used for the generator because it is suitable for dealing with time series data. The generator takes the input(x1, x2, … xn) of m features in the recent n timesteps and outputs the predicted price of the next timestep. The generator contains three GRU layers and is followed by three fully connected layers and the output G(x) can be defined as:

$$z_t = \sigma g(W_z * x_t + U_z * h_{t-1} + b_z)$$

$$r_t = \sigma g(W_r * x_t + U_r * h_{t-1} + b_r)$$
$$\hat{h_t} = \varphi h(W_h * x_t + U_h * (r_t * h_{t-1}) + b_h)$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \hat{h_t}$$

where W, U and b are parameters, ϕh refers to tanh, rt is the reset gate vector, h^t is the candidate activation vector, σg refers to sigmoid activation function, zt is the update gate vector, xt is the input and ht is the output.

Discriminator:

The discriminator takes the input of both real prices and the real prices combined with the predicted price of the generator. discriminator is expected to distinguish whether the input data contain prices generated by the generator. For example, if the input contains the generated price, it should output 1, and if the input does not contain, it should output 0. A CNN with 1d convolutional layers and fully connected layers is selected to be the discriminator. The price data from step 1 to step n+1 is used as the real data and the fake data is produced by concating real data from step 1 to step n and the predicted price for step n+1 from the generator.

4.1.2 Transformer

Transformer is a model based on the encoder-decoder structure, which discards the RNN in the previous seq2seq model and adopts Mulit-head-self-attention, allowing the input data to be processed in parallel and improving the operational efficiency. Our prediction model is based on the original Transformer structure.

Encoder:

The encoder consists of an input layer, a position encoding layer, and a stack of four identical encoder layers. Each layer consists of two sub-layers, a multi-head self-attention mechanism and a fully connected feed-forward network, where each sub-layer has a residual connection and normalisation, so that the output of the sub-layer can be represented as :

$$sublayer_{output} = LayerNorm(x + \big(SubLayer(x)\big))$$

The input layer maps the input time series data to the dimensional model vector through a fully connected network. This step is crucial for the model to adopt the multi-head attention mechanism. multi-head attention is the projection of Q, K, V by h different linear transformations and finally the stitching together of the different attention results.

$$MultiHead(Q,K,V) = Concat(head_1, \dots, head_h)W^O$$

Transformer discards RNN structure, whose biggest advantage is the abstraction of data on time series, so the relative position information is added using the Positional Encoding approach. That is, the sequence information in the time series data is encoded by adding an input vector and a position encoding vector using position encoding with sine and cosine functions. The resulting vectors are fed into four encoder layers.

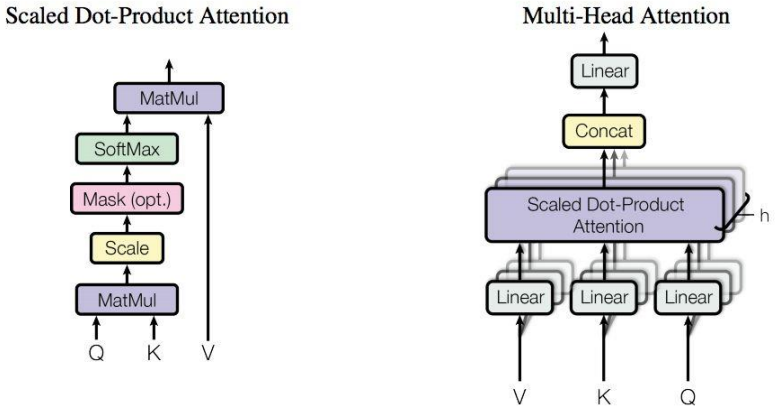


Fig. 2. Scaled dot-product multi-head attention (Vaswaniet et al., 2017)

Decoder

The input to the decoder section is the output generated by the previous time step, which is passed through the output plus the position information into the decoder, which consists of an input layer and an output layer. The decoder input starts from the last data point of the encoder input. The input layer maps the decoder input to a dimensional vector of the d-model. The final output layer, which is a simple fully connected layer via a Sigmoid activation function, maps the final output of the decoder to a y-value, i.e., the forex price,.

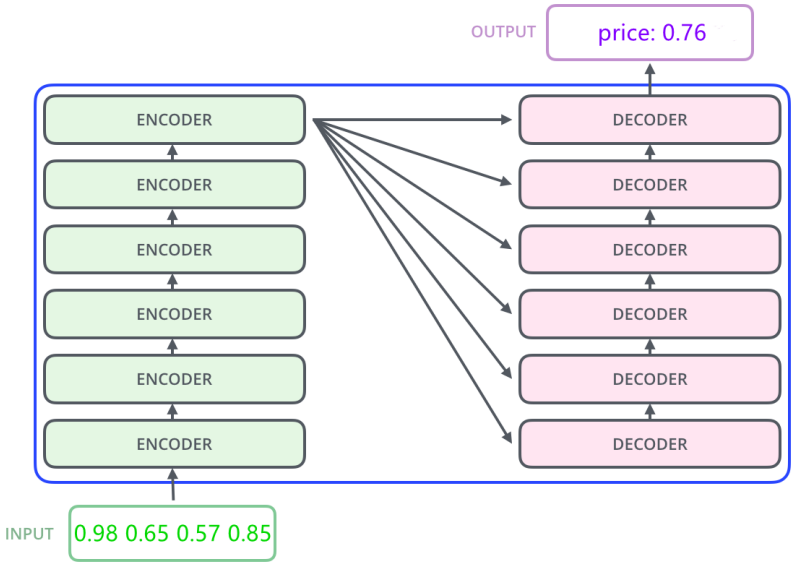


Fig. 3. Transformer architecture (Vaswaniet et al., 2017)

Positional Encoding

In the data preprocessing part of the Transformer, since the model does not have access to the input order of the time series as RNNs do, in order to deal with this problem, the transformer adds an additional vector Positional Encoding to the inputs of the encoder and decoder layers, with the same dimensionality as The dimension of embedding is the same, and for time step t and position i in the sequence, each input is given an embedding based on whether it is on an odd or even index in the sequence. The computation is as follows.

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{1000^{\frac{2i}{d_{model}}}}\right)$$

$$PE_{(pos,2i+1)} = \cos\left(\frac{pos}{1000^{\frac{2i}{d_{model}}}}\right)$$

The whole process is as follows.

1: Enter Multi-head Self-Attention, which hides future information and focuses only on the current sequence information

2: Add & Normalization process

3: Enter the Multi-head Attention component, where the input is the final output K and V of the encoder, and the vector obtained from the normalization process in the previous step.

4: The same Add & Normalization as Encoder, feedforward neural network, Add & Normalization process.

5: Mapping the final output of the decoder to the target time series with the linear layer.

4.1.3 Temporal Convolution Network

Convolutional networks have been shown to be effective in extracting high-level features from structured data, and temporal convolutional network is a neural network model that utilizes causal convolution and dilated convolution, which can adapt to the temporal nature of time series data and can provide receptive fields for temporal modeling.

Causal convolution

Since the sequential problem is to be handled, i.e., the temporal problem is to be considered, the normal CNN convolution cannot be used and a new CNN model must be used, in which causal convolution is used. The algorithm for the convolution is shown below, based on x1..... .xt and y1 .... .yt-1 to predict yt so that yt is close to the actual value.

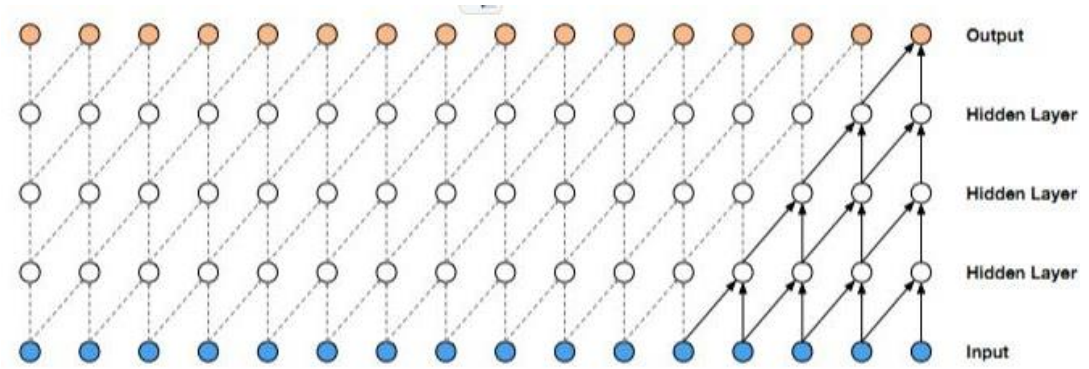


Fig. 4. Causal convolution (Bai et al., 2018)

Dilated convolution

For causal convolution, one of the problems is that many layers or a very large filter are needed to increase the field of perception of the convolution. In TCN, we increase the field of perception by enlarging the convolution with size alignment. Dilated convolution is the process of applying a filter to a region larger than the length of the filter itself by skipping part of the input. The illustration of dilated convolution is as follows：

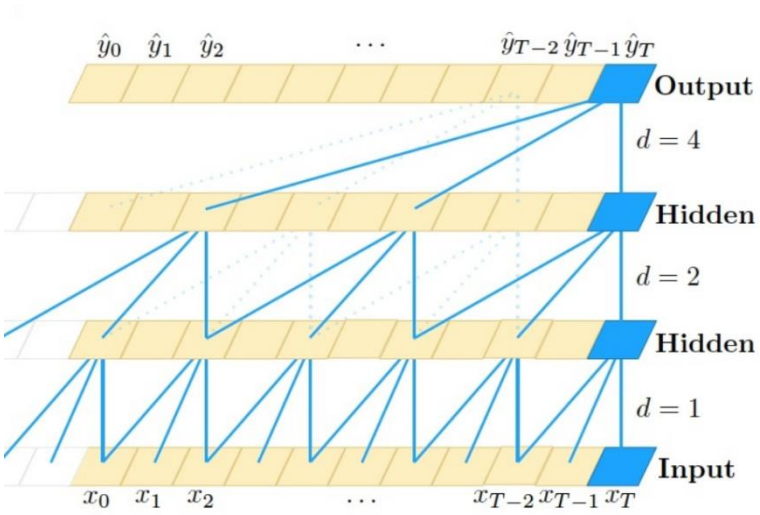


Fig. 5. Dilated convolution (Bai et al., 2018)

The equation for the dilated convolution is as follows:

$$F(s) = \sum_{i=0}^{k-1} f(i) * x_{s-d*i}$$

Residual Connect

Residual connections have proven to be an effective method for training deep networks, which allows the network to pass information in a cross-layer fashion.

The TCN model constructs a residual block to replace the convolution of one layer. As shown above, a residual block contains two layers of convolution and nonlinear mapping, and WeightNorm and Dropout are added in each layer to regularize the network.

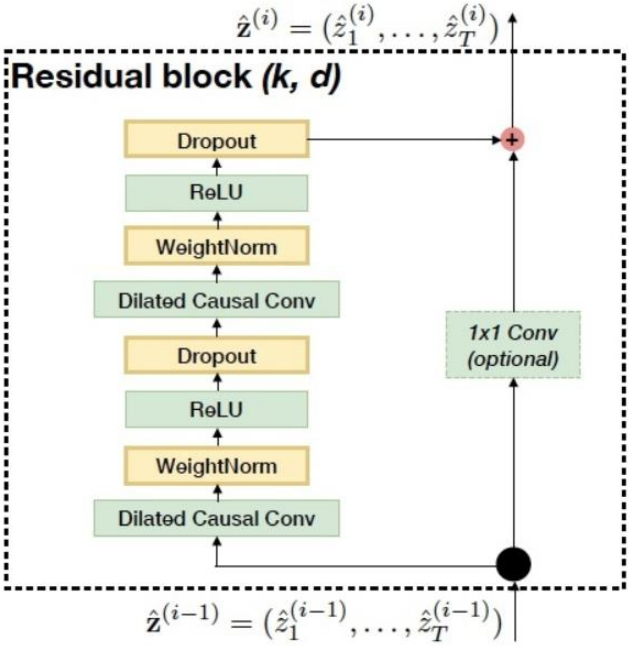


Fig. 6. Residual connect (Bai et al., 2018)

4.1.4 CNN+Bi-LSTM+Attention

Convolutional Neural Network (CNN) is a commonly used machine learning method, which consists of a solutional layer, padding, pooling layer, dropout, and fully connected layer. Each of these convolutional layers learns more features of the input object by back propagation, thus reducing the amount of data and increasing the learning efficiency.

The basic neural network architecture is as follows：

$$z_i = \sum_j W_{i,j} * X_j + b_i$$

In the equation, **z** refers to the output of the neural network, **w** is the weight, **x** is the input to the neural network, and **b** is the bias.

In addition, long short-term memory (LSTM) is also a very popular machine learning method, which is a variant of recurrent neural network (RNN). The purpose of LSTM is to solve the problem that traditional RNNs cannot handle long sequence of temporal

data effectively. The most special thing about LSTM is that it has three gates, which are forgetting gate, memory gate and output gate.

Each gate has an important role. Forget gate is an important step for lstm to learn the input features and it can decide whether to continue transferring the information from previous step to the next step. The function is as follows:

$$f_t = \sigma\big(W_f \cdot [h_{t-1}, x_t] + b_f\big)$$

In this function, $\boldsymbol{h_{t-1}}$ refers to previous output.

The memory gate differs from the forget gate in that it is used to determine what information is stored in the cell state. sigmoid and tanh layers are the components that make it up. sigmoid layer is used to update the learned features, while tanh layer is used to replenish the discarded information. The function is as follows:

$$i_t = \sigma(W_t \cdot [h_{t-1}, x_t] + b_i)$$

$$C_t = tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$$

In this function, i_t refers to the information that is updated and C_t refers to the information to be filled.

The output gate determines which data in the cell state needs to be output by using the sigmoid function. The function is as follows:

$$O_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$$

$$h_t = O_t * tanh(C_t)$$

The attention mechanism is a way to obtain effective information from a large amount of data by mimicking the conscious behavior of humans. In other words, it is a shift from focusing on all information to focusing only on the main points, thus increasing efficiency and obtaining effective information quickly. By using a combination of these three methods, the strengths of each can be fully exploited to achieve better performance.

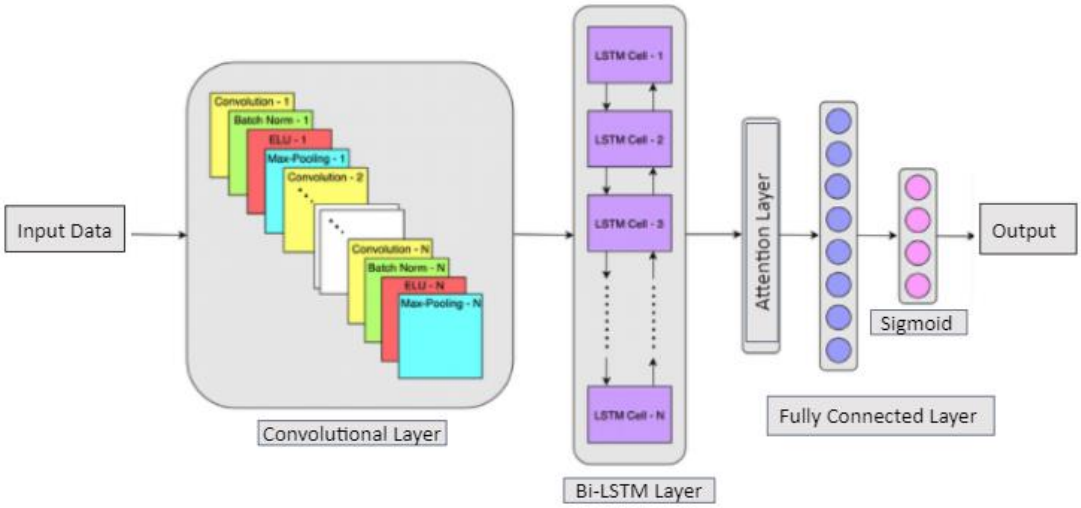


Fig. 7. CNN-LSTM-Attention

4.1.5 New Ideas

We have tried three different noval ideas on our base models described in above sections. The first two methods were suggested by Dr Khushi, which are popular in the industry. The third method is created by our project team.

1. Row Wise Scaling

We performed min max scaling in each observation window during training and testing, which means the scalar will be different when we move the observation window to next time stamp

2. Multiple Input

we performed modelling using dataset with three different time period (15-min, 30-min and 1 hour), then we concatenated the models to go through a fully connected layer to have the final prediction

3. Boosting

We borrowed the ideas from boosting. We proposed two boosting methods. The first one is to retrain the model. We trained the model with all training data first, then we made predictions on the training data and calculated the corresponding RMSE for

each prediction. Here we obtained the RMSE associated with the training set at all time stamps. Lastly, we looked back and picked up the training sets with RMSE higher than average, which is where the model performs badly. And we used those training sets to train the model again. Finally, we stacked two models together and went through a fully connected layer.
The second one is fitting the residuals. We trained the model with all training data, and then we used the model to make the prediction and calculate the corresponding residuals. We used those residuals as response variables and built another model to predict the residuals. Hence in back testing, we can use the residual predictions to adjust the next close price prediction, or we can adjust the trading strategy accordingly.

4.2 Data Collection

The Forex time series dataset can be collected from various sources/websites. In our project, the Forex dataset was produced by our tutor Ling, and she extracted the dataset from Oanda.com.

Oanda is a global leader in online multi-asset trading services, and it is one of the biggest forex trading brokers in the world. The dataset was downloaded by using the API function which Oanda produced.

The dataset contains the OHLC value of each currency pair. Each currency pair has five different time intervals: daily data, four-hour data, one-hour data, 30 minutes data, and 15 minutes data.

The timeframe was from 2005/01/02 to 2021/07/30. The AUD/CAD, AUD/NZD, EUR/AUD, EUR/USD have been used as our training and testing currency Pair.

4.3 Data Analysis

After we imported our dataset, we used Talib to calculate the MA, MACD, 20SD, upper band, lower band, EMA, CCI, Bollinger Bands, SMA, ADX and RSI of the dataset to increase the features of our dataset. We have used data scaling and dimension reduction to make our input dataset align with the same range. Also, we have used the

Keras functional API to merge the dataset with different time frames to get the optimal predicted result and try to use the row-wise scaling to form the dataset from 0 to 1. We used the RMSE to evaluate the model, and through the backtesting result, we calculated the trade win/lose rate with the daily profit.

4.4 Evaluation

4.4.1 Mean Square Error

We use MSE from the torch package as the loss function during the training and evaluation process, which is defined below:

$$\boldsymbol{MSE} = \frac{\mathbf{1}}{\boldsymbol{n}}\sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} \quad (\boldsymbol{y_i} - \hat{\boldsymbol{y}})^{\mathbf{2}}$$

4.4.2 Root Mean Square Error

RMSE is the square root of MSE, which can be easily calculated.

$$\boldsymbol{RMSE} = \sqrt{\frac{\mathbf{1}}{\boldsymbol{n}}\sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} \quad (\boldsymbol{y_i} - \hat{\boldsymbol{y}})^{\mathbf{2}}}$$

4.4.3 Mean Absolute Error

MAE is an arithmetic average of the absolute errors.

$$\boldsymbol{MAE} = \frac{\mathbf{1}}{\boldsymbol{n}}\sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} \quad |\boldsymbol{y_i} - \hat{\boldsymbol{y}}|$$

4.5 Backtesting

Our final trading strategy is presented below:

- buy signal: last close price is higher than current price and (prediction - buy buffer) is higher than current price
- sell signal: last close price is lower than current price and (prediction + sell buffer) is lower than current price
- otherwise: hold

In addition if we receive consecutive buy/sell signal and the current P/L is positive:

- increase profit taken by 2%
- increase stop loss by 50%

5. RESOURCES

Indicate the resources that will be required to complete your projects. Depending on the nature of the project, include sections where appropriate.

5.1 Hardware & Software

Hardware	
CPU	Intel Corei7 8700K 3.7GHz
RAM	16GB
GPU	NVIDIA RTX2070S
System	Windows 10
Software	
Environments	py 3.8 on Jupyter Notebook and Google Colab
Packages & Tools	Pytorch, Tensorflow, Scikit-Learn, Numpy, Pandas, Ta-lib

Table 1. Hardware and software resources

5.2 Materials

PyTorch, Keras, Pandas, Numpy and etc. from Python packages have been applied in our modelling. We have also learnt from the codes in Github and any publications can be found in Google or Google Scholar.

5.3 Roles & Responsibilities

Team Members	**Role**	**Role Description**
Shunfei Zhang	Project Manager	Responsible for guiding the team throughout each week, ensuring each member is

		keeping up with their responsibilities. setting up the team meeting and making the final decision.
Jialin (Chris) Li & Zhijun Xing	Machine Learning Engineer	In charge of acquiring the relevant information on the models. Afterwards, be responsible for the design and implementation of the deep learning models. Lastly, being in charge of trading strategy optimisation and backtesting.
Dezheng Fang & Bingtao Zhao	Data Analyst	Responsible for data cleaning inspection and pre-processing. Also in charge of general deep learning models on the given stock data to ensure the performance.

Table 2. Roles and responsibilities

6. MILESTONES / SCHEDULE

Milestone	Tasks	Reporting	Date
Week-1	Literature review and study basic knowledge of forex trade.	The knowledge learned in the research.	22/08/2021
Week-2	Continue research on forex trade and and model	Project objectives and scope in client meeting	22/08/2021
Week-3	Continue model implementation	Individual report	29/08/2021
Week-4	Conduct backtesting for models	Individual report	05/09/2021
Week-5	Proposal Report Due Working on the proposal report	Individual report and project proposal report	12/09/2021

Week-6	Continue to work on the improvement of the model.	Individual report and model review meeting	19/09/2021
Week-7	Design models for predicting highest high and lowest low.	Individual report and model review meeting	26/09/2021
Week-8	Continue model optimisation and backtesting.	Client meeting and presentation of proposed model	10/10/2021
Week-9	Progress Report and client meeting. Continue model optimisation and backtesting. Start to work on the final report.	Individual report Second client meeting	17/10/2021
Week-10	Come up with some novel ideas. Continue working on the final report. Apply existing models to more currencies.	Mid-semester project status check, individual report, and client meeting reporting	24/10/2021
Week-11	Continue brainstorming for novel ideas. Continue working on model optimisation.	Individual report and model review meeting	31/10/2021
Week-12	Start to prepare for presentation and finalize models	Individual report	07/11/2021
Week-13	Prepare for final presentation and live Q&A session	Individual report	14/11/2021
Week-14	Presentation Writing the final report	Q & A session with the client	21/11/2021
Week-15	Working on Final Report and models	Final Report	28/11/2021

Week-16	Organize code and models	Project Artifact	05/12/2021

Table 3. Schedule and milestones

7. RESULTS

In this project, we have tested more than four pairs of currencies on each of the four models and record the best performance on each currency pair. The results include out-of-sample evaluation results and back-testing results. Besides the four base model, we have also implemented three ideas. The results with some improvement will be shown in the section below.

7.1 GAN

	Currency	MSE	RMSE	MAE
GAN	AUD/CAD	2.53e-06	0.001591	0.001178
GAN	AUD/NZD	6.09e-06	0.002469	0.001920
GAN	EUR/USD	1.14e-05	0.003381	0.002683
GAN	EUR/AUD	4.16e-06	0.002077	0.001374
GAN +Row-wise scaling +Stacking +Boosting	EUR/AUD	3.27e-06	0.00181	0.001034

Table 3. GAN testing result

	Currency	Win Count	Lose Count	Win/Lose	P/L	Profit Per Day
GAN	AUD/CA	200	136	1.47	1131	6.2 pips

	D				pips	
GAN	AUD/NZD	271	225	1.20	799 pips	4.4 pips
GAN	EUR/USD	129	154	0.84	480 pips	2.6 pips
GAN	EUR/AUD	406	321	1.26	1781 pips	9.8 pips
GAN +Row-wise scaling +Stacking +Boosting	EUR/AUD			0.77	3231 pips	17.9 pips

Table 4. GAN best backtesting result

The GAN model archives lowest RMSE loss on the AUD/CAD and in the backtesting stage the model is most profitable on EUR/AUD and the introduction of the three techniques makes a positive impact on the model. The RMSE loss dropped 2e-04 and the profit per day increased 8 pips.

7.2 Transformer

Currency	MSE	RMSE	MAE
AUD/CAD	1.32E-06	0.001149	0.000828
AUD/NZD	1.44E-06	0.001198	0.000886
EUR/AUD	4.05E-05	0.006398	0.005043
EUR/USD	2.09E-05	0.00457	0.003597

Table 5. Transformer testing result

Transformer achieved lowest out-of-sample RMSE on AUD/CAD

Currency	Win Count	Lose Count	Win/Lose	P/L	Profit Per Day
AUD/CAD	610	501	1.22	740 pips	2.4 pips
AUD/NZD	194	142	1.37	217 pips	0.7 pips
EUR/AUD	332	191	1.74	3100 pips	9.9 pips
EUR/USD	66	89	0.74	500 pips	1.5 pips

Table 6. Transformer best backtesting result on each currency

However, in back-testing, Transformer achieved the best result on EUR/AUD with 9.9 pips per day and 1.74 win trade/lose trade ratio. Unfortunately, the transformer cannot work well with any of the three ideas described in the section above.

7.3 TCN

Currency	MSE	RMSE	MAE
AUD/CAD	5.65E-07	0.000752	0.000486
AUD/NZD	4.42E-07	0.000665	0.000439
EUR/AUD	2.77E-06	0.001663	0.000977
EUR/USD	9.22E-07	0.000959	0.000716

Table 7. Plan TCN testing results

TCN achieved lowest out-of-sample RMSE on AUD/NZD, and it outperformed Transformer on this currency.

Currency	Win Count	Lose Count	Win/Lose	P/L	Profit Per Day
AUD/CAD	229	120	1.91	1411 pips	4.5 pips
AUD/NZD	368	203	1.81	1236 pips	4.0 pips
EUR/AUD	900	827	1.09	2400 pips	7.7 pips
EUR/USD	382	791	0.48	100 pips	3.2 pips

Table 8. Plain TCN best back testing result on each currency

Like Transformer, TCN achieved best back-testing results on EUR/AUD with 7.7 pips per day and only 1.09 win trade/lose trade ratio. Compared with Transformer, TCN is not satisfying. If we look at win trade/lose trade ratio, TCN has 1.91 on AUD/CAD and 1.81 on AUD/NZD. However, the profit is not too high, with less than 5 pips per day.

In addition, TCN works well with multiple inputs. We can see the RMSE has increased with multiple inputs.

Model	Currency	MSE	RMSE	MAE
TCN	EUR/AUD	2.77E-06	0.001663	0.000977
TCN Multiple Inputs	EUR/AUD	3.85E-06	0.00623	0.005297
TCN	AUD/CAD	5.65E-07	0.000752	0.000486
TCN Multiple Inputs	AUD/CAD	1.34E-06	0.001159	0.000764
TCN	AUD/NZD	4.42E-07	0.000665	0.000439
TCN Multiple Inputs	AUD/NZD	1.41E-06	0.001187	0.000979

Table 9. Plan TCN VS TCN with multiple inputs

In back-testing, TCN can achieve much better profits with multiple inputs with 16.2 pips per day. However, the win trade/lose trade ratio is still 1.11. On AUD/CAD TCN achieved slightly higher profit with multiple inputs, but the ratio drops to 0.93. On AUD/NZD, TCN can still maintain a positive profit, but performance is hugely reduced by multiple inputs.

Model	Currency	Win Count	Lose Count	Win/Lose	P/L	Profit Per Day
TCN	EUR/AUD	900	827	1.09	2400 pips	7.7 pips
TCN Multiple Inputs	EUR/AUD	735	665	1.11	5049 pips	16.2 pips
TCN	AUD/CAD	229	120	1.91	1411 pips	4.5 pips
TCN Multiple Inputs	AUD/CAD	717	767	0.93	1535 pips	4.9 pips
TCN	AUD/NZD	368	203	1.81	1236 pips	4.0 pips
TCN Multiple Inputs	AUD/NZD	159	182	0.87	257 pips	0.8 pips

Table 10. Plan TCN VS TCN with multiple inputs (back teting)

7.4 CNN+Bi-LSTM+Attention

Currency	MSE	RMSE	MAE
AUD_CAD	1.36e-4	1.1657e-2	1.0465e-2
AUD_NZD	1.5e-5	3.915e-3	3.104e-3
EUR_AUD	7.19e-4	2.6806e-2	2.2266e-2
EUR_USD	2.43e-4	1.5577e-2	1.2327e-2

Table 11. CNN+Bi-LSTM+Attention testing result

For RMSE, the model performs best at AUD_NZD with 3.915e-3. Conversely, the worst is at EUR_AUD data with 2.6806e-2.

Model	Currency	Win Count	Lose Count	Win/ Lose	P/L	Profit Per Day
CNN+Bi-LSTM+Attention	AUD_CAD	258	250	1.032	104 pips	0.35 pips
CNN+Bi-LSTM+Attention	AUD_NZD	233	206	1.131	319 pips	1.06 pips
CNN+Bi-LSTM+Attention	EUR_AUD	353	430	0.821	1568 pips	5.23 pips
CNN+Bi-LSTM+Attention	EUR_USD	214	235	0.911	58 pips	0.2 pips

Table 12. CNN+Bi-LSTM+Attention best backtesting result on each currency

In the model profitability, the best performance in EUR_AUD data, with an average of 5.23 pips per day, but its RMSE performance is not good, which will be explained in the following discussion. On the overall view, the best performance is AUD_NZD currency pair, with an average daily profit of 1.06 pips and a win/loss ratio of 1.131.

7.5 Model Comparison

In this section, we presented the model by currency pairs. In this way, we can compare the model performance on the same pair of currency. LSTM+CNN+Attention generally achieved the worst results.

7.5.1 AUD/CAD

LSTM+CNN+Attention clearly underperformed on RMSE. GAN achieved highest profit per day while TCN has the highest win/loss ratio.

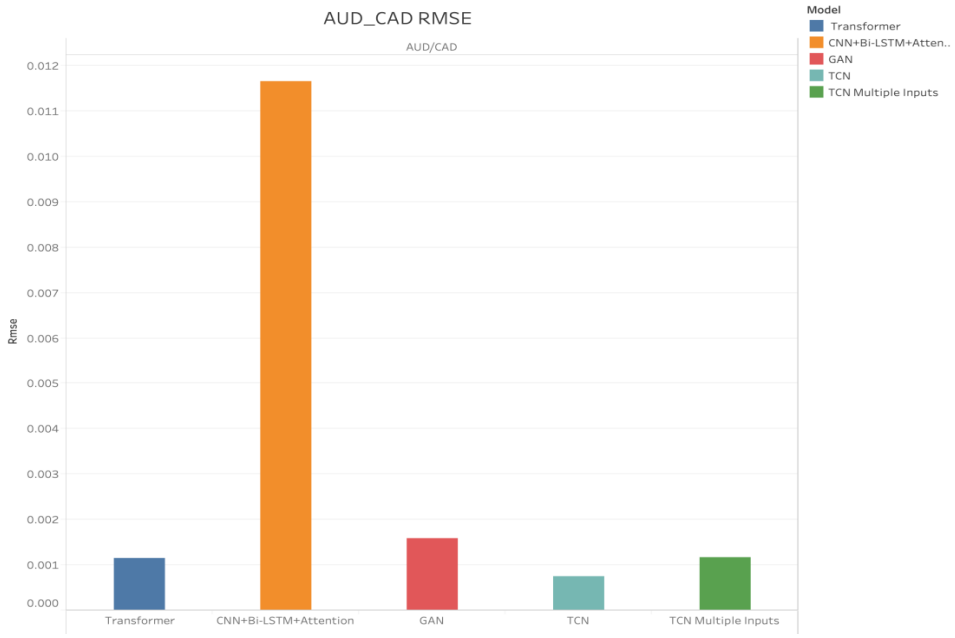


Fig. 8. Best RMSE of each model on AUD/CAD

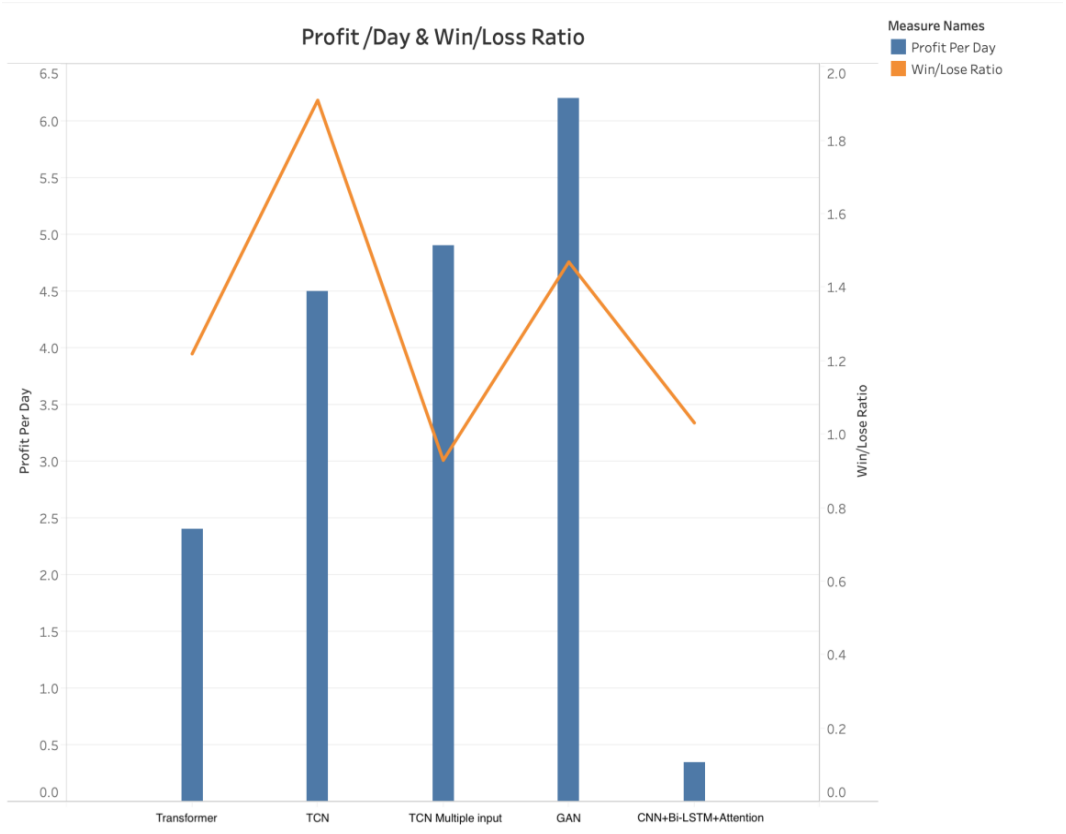


Fig. 9. Best backtesting performance of each model on AUD/CAD

7.5.2 AUD/NZD

On AUD/NZD, GAN didn't have a good RMSE. Both TCN and GAN can achieve over 4 pips profit per day, while TCN has a better win/loss ratio. Although GAN has much higher RMSE than TCN, it achieved a better back-testing result, which means better RMSE doesn't guarantee a better back-testing result.

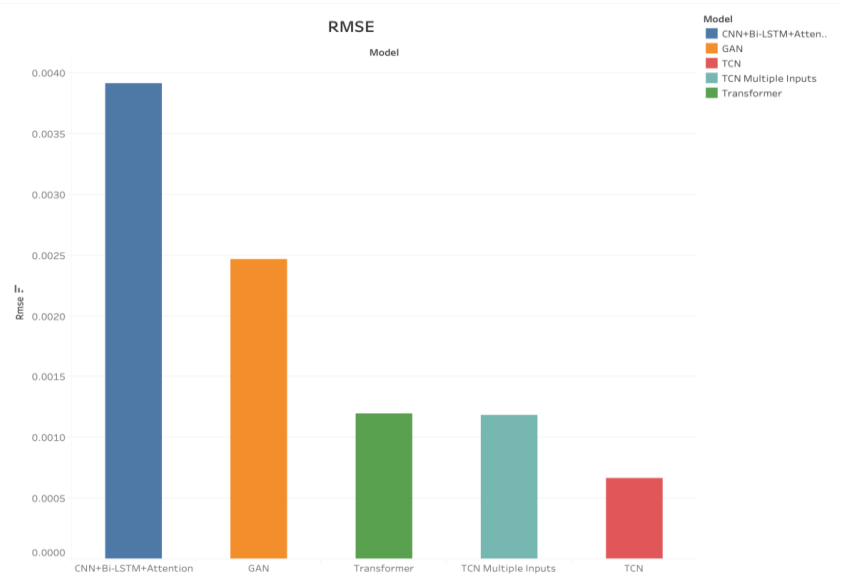


Fig. 10. Best RMSE of each model on AUD/NZD

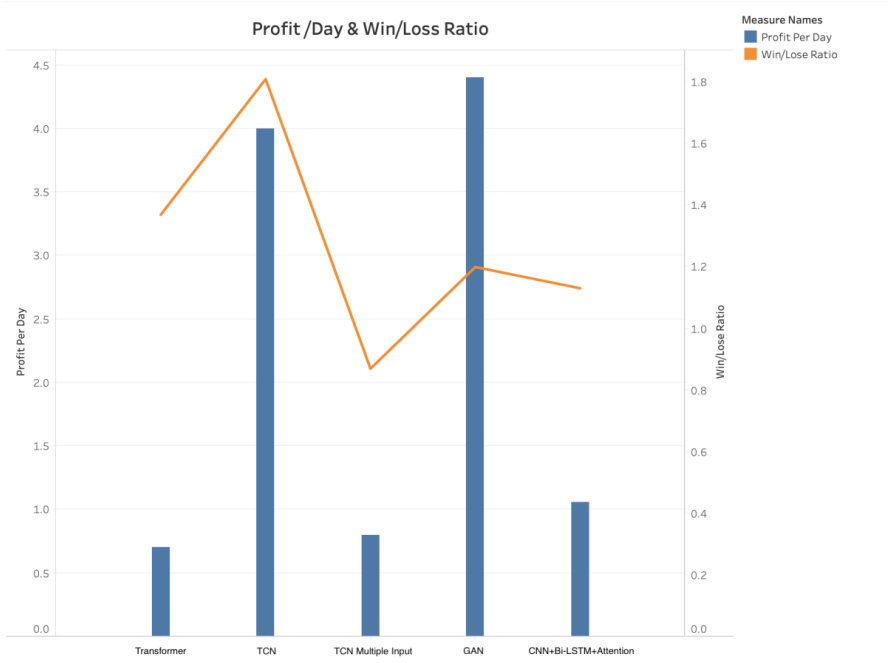


Fig. 11. Best backtesting performance of each model on AUD/NZD

7.5.3 EUR/AUD

TCN and GAN clearly have the lowest RMSE. TCN with multiple input achieved the highest profit per day (16 pips), but the win/loss ratio is just above 1. Again although TCN with multiple input has relatively higher RMSE, but a better profit.

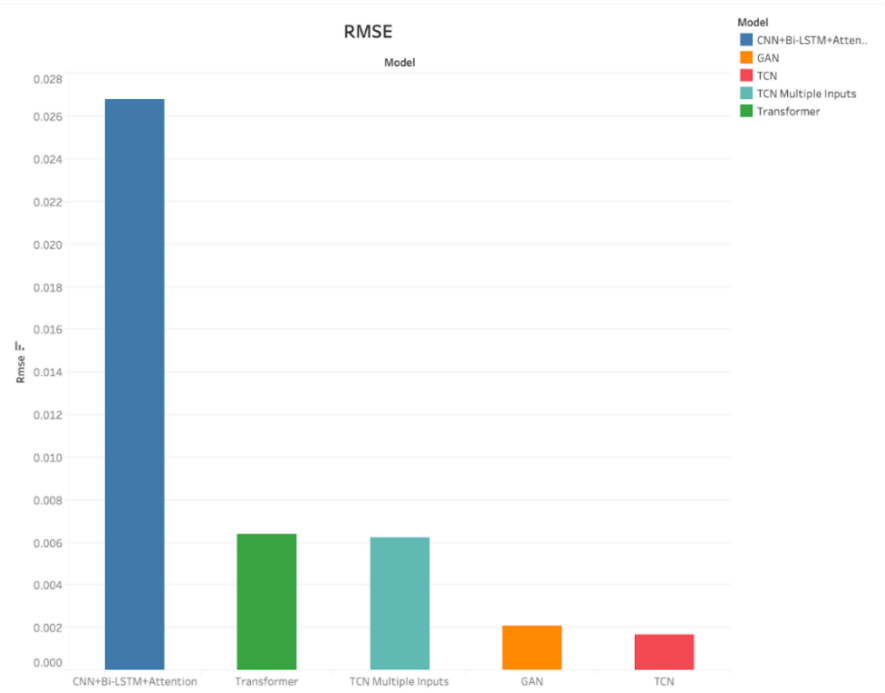


Fig. 12. Best RMSE of each model on EUR/AUD

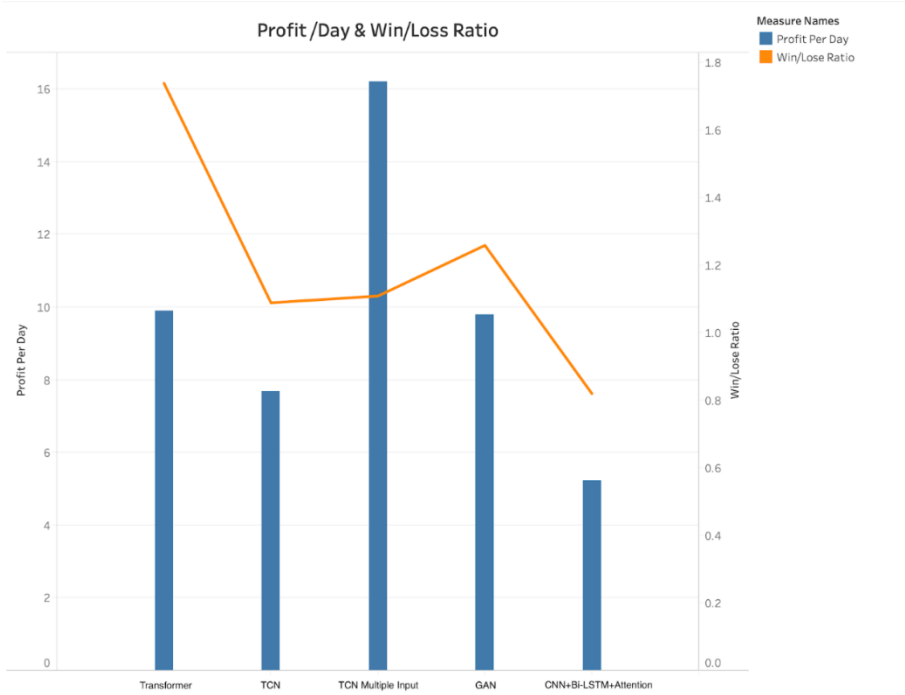


Fig. 13. Best backtesting performance of each model on EUR/AUD

7.5.4 EUR/USD

TCN clearly outperformed other models on RMSE. All four models performed badly on EUR/USD, GAN has the highest profit per day recording only 2.6 pips. The win/loss ratios for all models are less than 1.

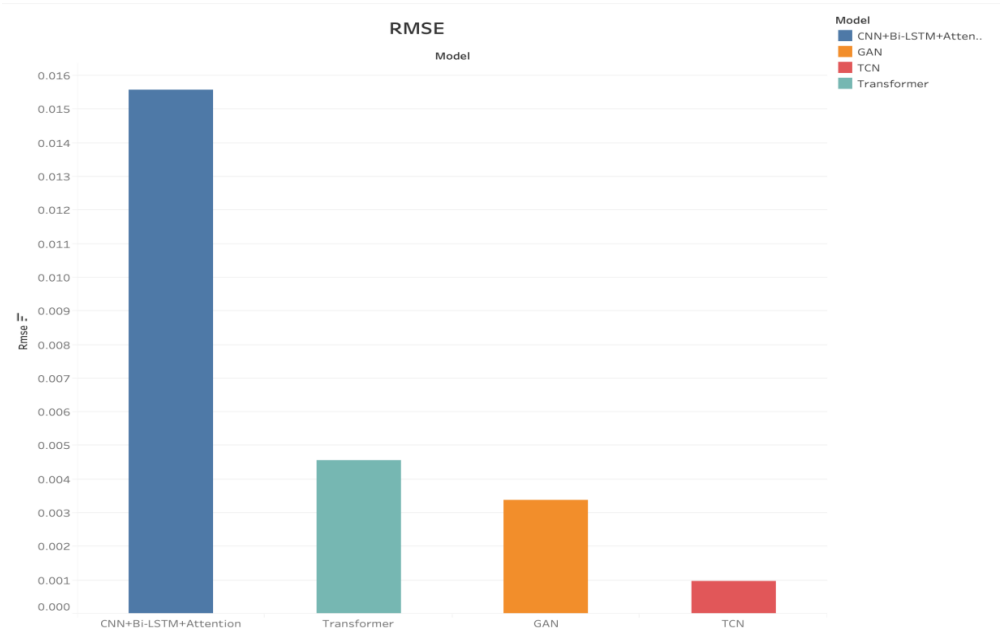


Fig. 14. Best RMSE of each model on EUR/USD

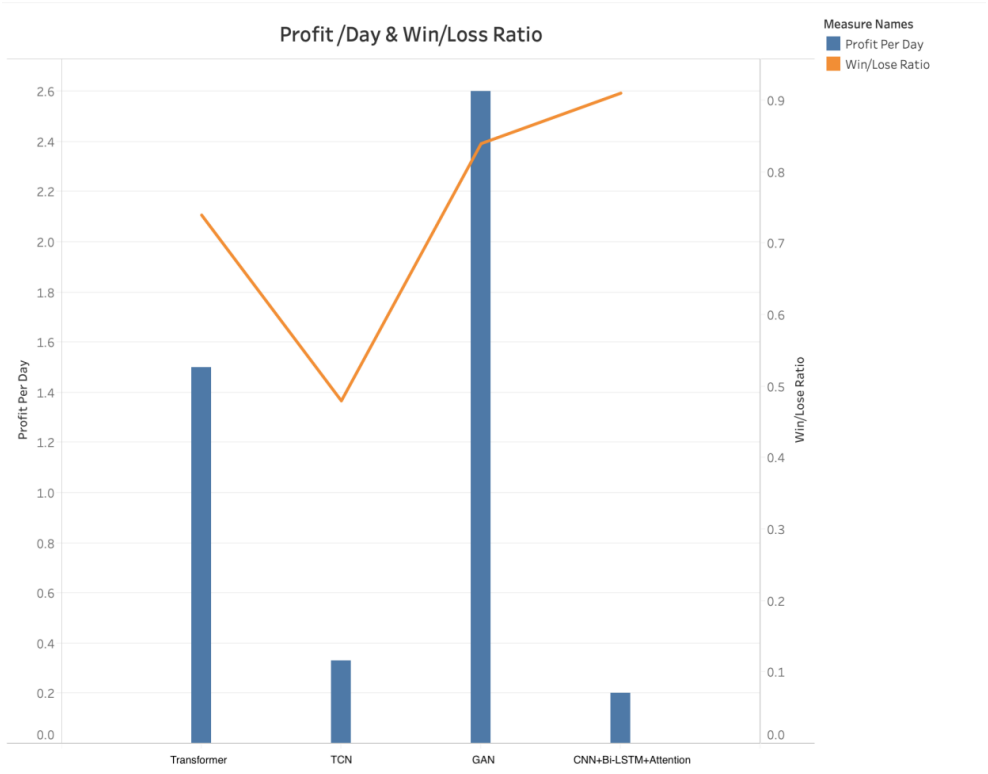


Fig. 15. Best backtesting performanceof each model on EUR/USD

8. DISCUSSION

The main goal of this project is to predict the closing price at 1 point in time in the future based on historical data. We used four relatively new deep learning models to predict the prices. They all performed relatively well in prediction, and in addition to that, we tried three methods for enhancement.

8.1 data pre-processing

The original dataset is OHLC data from a certain period without any other features and data. For feature engineering, we first calculated about 15 or so features as input, such as EMA, SMA, and Bollinger Bands. All these features were computed using TA-Lib based on OHLC values, so they are all relevant to our objective.
At the beginning, we did not include these technical indicators as inputs and the model showed poor performance results. We found that the output of the model had values close to the last point, i.e., it was horizontally shifted, because the output was too highly linearly correlated with the close price of the inputs due to too few features of the model. This issue is mitigated by the addition of technical indicators and other features.

8.2 GAN

The GAN model achieves profitability in all currency pairs. It achieves rather high pips per day on EUR/AUD but the win/lose ratio is lower than average, which means the model's prediction performance is not stable. Another problem of GAN is its training efficiency. The training of the gan models is extremely time consuming as the loss of the gan can not be an indicator for convergence. One way for achieving better results is to run a large number of epochs and select the best weight depending on the loss regardless of the extra cost in time and computing resources. We also tried to use WGAN and WGAN with gradient penalty. However, these attempts are all failed and the results are discarded as the results of both two are much worse than the GAN despite the advantage of meaningful training loss trajectory.

8.3 Transformer

According to our experimental results in the previous section, the Transformer model is the second-best model in terms of RMSE among all our models. We add two Bi-LSTM layers to capture the sequence information before going through the encoder layer, but this may not provide a better prediction result. We also try to use positional encoding, which is widely used in NLP tasks, i.e., the relationship between two positions used in the original text can be modelled by affine transformation between their positional encodings. The use of sine and cosine functions with different frequencies as positional encoding may not be applicable to our context, since we use features from the same time step.

Another observation from the experiments is that the number of Dmodels and heads and the size of the key and value vectors do not have a dramatic effect on the performance of the Transformer model. When the number of heads is larger and the size of key, value is larger, more model parameters are required. This causes an increase in training time, but does not dramatically affect the RMSE, so we can use the appropriate values for training. In the training process, we use early stop to prevent overfitting, and stop training when the loss of the validation set does not continue to drop for 8 epochs and obtain the best model results.

8.4 TCN

According to our experimental results in the previous section, the TCN model has the best prediction results among all our models, outperforming all models, including the baseline model LSTM. and the training results are more stable due to the excellent performance of causal convolution in such type of input of time series and the method of inflated convolution allows the model to have a very large receptive fields with a small number of layers.

However, we observe that the prediction results of the TCN model do not perform as well as Transformer and GAN for some currency pairs in backtesting, i.e., a smaller RMSE does not lead to more substantial gains. But compared to other models, TCN's backtesting earnings have been the most stable after we saw the return curve. This

represents that TCN's prediction results have a lower risk of loss despite not leading to high returns.

8.5 CNN+BiLSTM+Attention

Among the four currencies, the model's RMSE value performs best on AUD_NZD data at 3.915e-3. In addition, EUR_AUD has the worst RMSE value at 2.6806e-2. For the model's return, the highest return is 5.23 per day despite the worst RMSE value for EUR_AUD. The rest of our group's models have a similar situation in EUR_AUD, so it may be because our trading strategy happens to fit the OHLC data of EUR_AUD, so there will be a situation where the model RMSE is not good, but the return is good. In addition, in terms of model optimization and improvement, I tried adding technical indicators, row-wise scaling, using functional API to implement multi-input models and using LSTM with three different hidden states for model averaging. However, the performance of the models did not improve, and some of them performed even worse. This confused me and leading me to wonder if there is something wrong with the structure of the model or if the implementation is correct. There is a lot of research on this stacking model that shows that there is no problem with the structure of the model. Therefore, I guess it is because the same model structure behaves differently for different data sets. In addition, adding too many methods to a proven model structure may cause performance degradation.

By comparing the other models, although CNN+BiLSTM+Attention achieved profitability for all four currency transactions, the profits were small. In addition, despite having been trying some methods to help improve the model performance, the model did not perform too much beyond the original model. This may be because I have not tried enough methods to find one that can improve the performance substantially.

8.6 Row-wise scaling

We can observe that when we encounter some prediction data value which is larger than the previous max value, then this value will be normalized to more than 1. To solve this problem, we try to scale the data within each window size.

We tried row-wise scaling on each of these four models to preprocess the data. On the TCN and Transformer models, this method leads to a non-convergence problem of the model, with the loss always remaining high during training. We believe that it may be because the use of min-max scale in each window size causes unbalanced distribution of the values of the data, which leads to the difficulty of convergence. Another point is that large fluctuations and small fluctuations will make all data in the value of 0-1, which makes the model difficult to judge.

In the other two models, only the maximum scale is used, and the model can be trained properly in this way. However, when some specific values (in the Technical indicator) in the data within a window size alternate between positive and negative, the scale is negative. After both models tried row-wise scaling, the prediction performance of the cnn-lstm-attention model became worse, so we discarded the method.

The final results indicate that this method works well on the GAN model, while the experimental results are poor on all three models, so we only use this method on the GAN model.

8.7 Combined 3 time period data

Some problems arise when we use data from only one time period. For example, When we use the 1-hour data to train the model, we will only have the information from OHLC; we don't know how the price changes during one-hour time. Hence we thought bringing in 30-min and 15-min data would help the model better capture the price trend and make a better prediction.

The final evaluation results indicated that the RMSE of most models using this method would be a bit higher than the initial model.This means that combining all temporal data overlaid together to predict does not produce better results.

8.8 Boosting

1.When we investigated the training performance, we saw the model performs badly when the close price has a huge jump or drop. Hence we were trying to find a way to deal with this issue.
With the idea of boosting, we thought the model will benefit if we pick up those poorly performing data and retrain the model. Finally, the result is output together with the original model. We experimented this method on the GAN model and found a small reduction in the evaluated RMSE, then we think this method is effective.

2.In the second boosting method we introduced, our initial intention was to introduce more flexibility to our trading strategy. With the idea of boosting, we came up with the idea that we use one model to predict close price, and then we use another model to predict residual. Together with the residual, we can either adjust the close prediction or adjust buy/sell buffer in the trading strategy. We experimented with this method on the TCN model, but the accuracy of the residual classification model is not very high, and using this method to determine the trading strategy only reduces the profit.
We first try to divide the values of residual into ten categories for training, and the final accuracy of the training set is about 27%, and the accuracy of the validation set is 40%, which is applied to the backtesting, i.eThe residual category is traded as a buffer. However, the effect is lower than the original fixed buffer value. This may be due to the fact that the accuracy of the test set is not high, which causes most of the buffer values to deviate too much from the expectation, resulting in worse results.
We divided the values of residual into dichotomous categories for training, i.e., only the positive and negative of residual were predicted and used to the positive and negative of the buffer. The final accuracy of the training set is about 57% and the accuracy of the validation set is 64%. We apply the classification results to the buffer, but with less impact on the results.

9. LIMITATIONS AND FUTURE WORKS

There are several limitations in the work of this project. The first limitation is the

9.1 Limitations

9.1.1 Time Constraint

The first constraint is that the project only undertook 13 weeks. In the first couple weeks, we spent some time reviewing the relevant literature, researching forex trading related financial knowledge, and replicating existing models. During the project, we also spent some time developing models to predict the highest high and lowest low price. However, the results were not satisfying, and we resumed to focus on prediction of the next close price. As a result, we didn't have enough time to develop a new model structure or to come up with more novel ideas to improve the existing models.

9.1.2 No Practical Application

The profitability of the models is not examined with real world trading. The models are now profitable in the back testing, but the real profit will definitely be smaller because the back testing environment is only a roughly simulated environment for the real forex market. The broker cost is set to be 2 pips in the back testing stage but in the real trading environment the transaction cost can be dynamic and, in many situations, will be larger than 2 pips. Besides, we set our back testing simulation based on the last recent year's records but the macro environment such as macro economy, international politics and global pandemic have significantly impacted most foreign currency rates. We are in a world of uncertainty. In the post-pandemic age, the moves in foreign currency rate may follow a different inner logic so that these back testing results might be less representative than expected.

9.1.3 Correlation

The correlation of some currency pairs and commodities are well known in professional foreign currency traders. We are not able to cover all currencies and commodities in the same model. Currently we only process and feed in these data separately and assume the independence among different currencies. We have not recognized the influence of one currency on others. There might be an opportunity to achieve better results by considering multiple currencies at the same time.

9.2 Future Work

In this step, we have only implemented our different model with our modified trading strategy but did not use our model to predict the forex value in the real trading platform. The real trading will be different from something we have done in the back testing result, so that the next step will be implementing our model to the real live trading.

Secondly, we have found the optimal trading buffer in our trading strategy for each different currency pair. But in live trading, the trading buffer within the trading strategy can be changed over time, which means we can add one reinforcement learning layer at the end of the model to help us decide the trading buffer range. This method will highly increase the win/lose trading rate, thus making more profit in live trading.

Thirdly, the latest machine learning research outcomes can be utilized in the financial prediction domain. These models or techniques with higher effectiveness and efficiency in other domains will be helpful to improve the performance in the AI trading task.

10. REFERENCES

Areekul, P., Senjyu, T., Toyama, H., & Yona, A. (2010). A Hybrid ARIMA and Neural Network Model for Short-Term Price Forecasting in Deregulated Market. Japan.

Bai, S., Kolter, J. Z., & Koltun, V. (2018). An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. arXiv preprint arXiv:1803.01271.

Box, G. E., Jenkins, G. M., Reinsel, G. C., & Ljung, G. M. (2015). Time series analysis: forecasting and control. John Wiley & Sons.

Ding, Q., Wu, S., Sun, H., Guo, J., & Guo, J. (2020). Hierarchical Multi-Scale Gaussian Transformer for Stock Movement Prediction. In IJCAI (pp. 4640-4646).

Eapen, J., Bein, D., & Verma, A. (2019, January). Novel deep learning model with CNN and bi-directional LSTM for improved stock market index prediction. In 2019 IEEE 9th annual computing and communication workshop and conference (CCWC) (pp. 0264-0270). IEEE.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2014). Generative adversarial nets. Advances in neural information processing systems, 27.

Gudelek, M. U., Boluk, S. A., & Ozbayoglu, A. M. (2017, November). A deep learning based stock trading model with 2-D CNN trend detection. In 2017 IEEE Symposium Series on Computational Intelligence (SSCI) (pp. 1-8). IEEE.

Huang, W., Nakamori, Y., & Wang, S. Y. (2005). Forecasting stock market movement direction with support vector machine. Computers & operations research, 32(10), 2513-2522.

Hu, Z., Zhao, Y., & Khushi, M. (2021). A survey of forex and stock price prediction using deep learning. Applied System Innovation, 4(1), 9.

Kim, T., & Kim, H. Y. (2019). Forecasting stock prices with a feature fusion LSTM-CNN model using different representations of the same data. PloS one, 14(2), e0212320.

Lin, Z., Jain, A., Wang, C., Fanti, G., & Sekar, V. (2020, October). Using GANs for sharing networked time series data: Challenges, initial promise, and open questions. In Proceedings of the ACM Internet Measurement Conference (pp. 464-483).

Lu, W., Li, J., Li, Y., Sun, A., & Wang, J. (2020). A CNN-LSTM-based model to forecast stock prices. Complexity, 2020.

Mnih, V., Heess, N., & Graves, A. (2014). Recurrent models of visual attention. In Advances in neural information processing systems (pp. 2204-2212).

Parida, A. K., Bisoi, R., & Dash, P. K. (2016). Chebyshev polynomial functions based locally recurrent neuro-fuzzy information system for prediction of financial and energy market data. The Journal of Finance and Data Science, 2(3), 202-223.

Patel, J., Shah, S., Thakkar, P., & Kotecha, K. (2015). Predicting stock market index using fusion of machine learning techniques. Expert Systems with Applications, 42(4), 2162-2172.

Siripurapu, A. (2014). Convolutional networks for stock trading. Stanford Univ Dep Comput Sci, 1(2), 1-6.

Tsai, M. C., Cheng, C. H., Tsai, M. I., & Shiu, H. Y. (2018). Forecasting leading industry stock prices based on a hybrid time-series forecast model. PloS one, 13(12), e0209922.

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

Wang, J., Wang, Z., Li, J., & Wu, J. (2018, July). Multilevel wavelet decomposition network for interpretable time series analysis. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (pp. 2437-2446).

Wen, Q., Yang, Z., Song, Y., & Jia, P. (2010). Automatic stock decision support system based on box theory and SVM algorithm. Expert systems with Applications, 37(2), 1015-1022.

Wu, N., Green, B., Ben, X., & O'Banion, S. (2020). Deep transformer models for time series forecasting: The influenza prevalence case. arXiv preprint arXiv:2001.08317.

Zeng, Z., & Khushi, M. (2020, July). Wavelet denoising and attention-based RNN-ARIMA model to predict forex price. In 2020 International Joint Conference on Neural Networks (IJCNN) (pp. 1-7). IEEE.

Zhang, K., Zhong, G., Dong, J., Wang, S., & Wang, Y. (2019). Stock market prediction based on generative adversarial network. Procedia computer science, 147, 400-406.

Zhou, X., Pan, Z., Hu, G., Tang, S., & Zhao, C. (2018). Stock market prediction on high-frequency data using generative adversarial nets. Mathematical Problems in Engineering, 2018.