mathematics

MDPI

Article

A Correlation-Embedded Attention Module to Mitigate Multicollinearity: An Algorithmic Trading Application

Jireh Yi-Le Chan [1,*,†], Steven Mun Hong Leow [1,†], Khean Thye Bea [1], Wai Khuen Cheng [2], Seuk Wai Phoong [3], Zeng-Wei Hong [4], Jim-Min Lin [4] and Yen-Lin Chen [5,*]

[1] Faculty of Business and Finance, University Tunku Abdul Rahman, Perak 31900, Malaysia; steven.utar@1utar.my (S.M.H.L.); beakheanthye@1utar.my (K.T.B.)
[2] Faculty of Information and Communication Technology, University Tunku Abdul Rahman, Perak 31900, Malaysia; chengwk@utar.edu.my
[3] Department of Operation and Management Information System, Faculty of Business and Accountancy, University of Malaya, Kuala Lumpur 50603, Malaysia; phoongsw@um.edu.my
[4] Department of Information Engineering and Computer Science, Feng Chia University, Taichung 40724, Taiwan; zwhong@fcu.edu.tw (Z.-W.H.); jimmy@fcu.edu.tw (J.-M.L.)
[5] Department of Computer Science and Information Engineering, National Taipei University of Technology, Taipei 106344, Taiwan
* Correspondence: jirehchan@utar.edu.my (J.Y.-L.C.); ylchen@mail.ntut.edu.tw (Y.-L.C.)
† These authors contributed equally to this work.

Abstract: Algorithmic trading is a common topic researched in the neural network due to the abundance of data available. It is a phenomenon where an approximately linear relationship exists between two or more independent variables. It is especially prevalent in financial data due to the interrelated nature of the data. The existing feature selection methods are not efficient enough in solving such a problem due to the potential loss of essential and relevant information. These methods are also not able to consider the interaction between features. Therefore, we proposed two improvements to apply to the Long Short-Term Memory neural network (LSTM) in this study. It is the Multicollinearity Reduction Module (MRM) based on correlation-embedded attention to mitigate multicollinearity without removing features. The motivation of the improvements is to allow the model to predict using the relevance and redundancy within the data. The first contribution of the paper is allowing a neural network to mitigate the effects of multicollinearity without removing any variables. The second contribution is improving trading returns when our proposed mechanisms are applied to an LSTM. This study compared the classification performance between LSTM models with and without the correlation-embedded attention module. The experimental result reveals that a neural network that can learn the relevance and redundancy of the financial data to improve the desired classification performance. Furthermore, the trading returns of our proposed module are 46.82% higher without sacrificing training time. Moreover, the MRM is designed to be a standalone module and is interoperable with existing models.

Keywords: algorithmic trading; multicollinearity; feature selection; neural network; classification

MSC: 68T07

check for updates

Citation: Chan, J.Y.-L.; Leow, S.M.H.; Bea, K.T.; Cheng, W.K.; Phoong, S.W.; Hong, Z.-W.; Lin, J.-M.; Chen, Y.-L. A Correlation-Embedded Attention Module to Mitigate Multicollinearity: An Algorithmic Trading Application. *Mathematics* **2022**, *10*, 1231. https://doi.org/10.3390/math10081231

Academic Editor: Christoph Frei

Received: 3 March 2022
Accepted: 6 April 2022
Published: 8 April 2022

1. Introduction

Algorithmic trading has become increasingly popular with the advancement of technologies and big data. It refers to any programmed software that automates one or more stages of the trading process [1]. It may contain hundreds of variables and minor changes that significantly affect forecast performance. As the forecasts are commonly conducted for trading, multicollinearity has vast implications on the system's profitability. Multicollinearity is an approximately linear relationship between two or more independent variables.

There are two significant problems of multicollinearity. Firstly, the estimated standard errors of the regression coefficient are large. Therefore, the parameter estimates are unreliable and decrease the precision [2]. Secondly, the model will have poor generalization capability and overfit the data. Financial data are likely to have high multicollinearity. For example, when using technical indicators in stock analysis, there will be multicollinearity issues if the indicators measure the same information, such as momentum [3]. This is because the different indicators are all derived from the same series of closing prices in such cases.

The technique to mitigate multicollinearity is feature selection, which can be categorized into the filter, wrapper, and embedded methods [4]. The feature selection method reduces the number of variables to the relevant few. The filter method uses the statistical properties of each feature and its relationship with the target variable to select the best subset of the variables. The examples are information gain, correlation, and chi-square. The wrapper method searches for a subset of features based on their performance. The subset is removed or added accordingly, but it is computationally expensive due to repeated learning. Examples of wrapper methods are forward selection and backward selection. Embedded methods are learning algorithms that perform feature selection in conjunction with the model tuning process. Examples include the Lasso Regression and Decision Tree. The advancement in machine learning and artificial intelligence has provided a new frontier of potential methods to improve performance. For example, ref. [5] proposed a feedforward artificial neural network to model data with multicollinearity and obtained much better performance in terms of the Root Mean Square Error (RMSE) compared to the traditional Ordinary Least Squares (OLS).

Existing feature selection methods are inadequate for a neural network because most statistical measures cannot uncover nonlinear relationships [6]. Furthermore, it is not optimal to remove features in financial data because they are all interrelated [7]. We propose using attention mechanisms and correlation embeddings on the input data to solve the multicollinearity issue in the financial forecast. This is an embedded approach to feature selection. The embeddings method developed for text data has successfully learned the similarity between words. The expectation is that the neural network embedding approach can learn the similarity between features. The model uses the correlation between variables as embedding. It gives the model information about redundancy between variables. The purpose of the attention mechanism is to learn which variables are most relevant to the target variable. It does this by assigning a different weight to each input variable. The attention model was first introduced for sequence-to-sequence machine translation by [8]. Today, the mechanism has continued to be developed and extended to other fields such as image data [9]. Attention allows the predictive model to learn the dependence between the variables and the output. The attention mechanism has proved to be useful in predicting the price fluctuations of stock markets using technical indicators [10]. The proposed method offers potential improvements in financial forecasting and algorithmic trading under the effects of multicollinearity.

We adopted the attention mechanism and correlation embeddings to a neural network that predicts price changes in foreign exchange rates. It is an embedded approach to feature selection. The first contribution of the paper is allowing a neural network to mitigate the effects of multicollinearity without removing any variables. The second contribution is improving trading returns when our proposed mechanisms are applied to an LSTM.

The rest of the paper is organized as follows. In Section 2, related works on multicollinearity in statistical and neural network models are presented. Section 3 discusses the methodology, where stock price data and the model are given. The detailed financial performance of the model is given in Section 4. Finally, we conclude in Section 5.

2. Related Works

2.1. Algorithmic Trading

Algorithmic trading is a process of trading financial instruments using preprogrammed systems. Algorithmic trading grew in popularity along with its technological advancement, and technical analysis has been used to make investment decisions in financial markets. Technical analysis uses chart patterns, price, and trading volume to predict the future price of assets. This information is used to form technical indicators on the asset's trend, momentum, volume, and volatility. The indicators give traders signals to enter or exit a trade. Besides that, fundamental data such as macroeconomic data and financial reports are utilized. This massive amount of data is subject to the effects of multicollinearity.

Forecasting of time series data is vital in the finance and economic field. It is traditionally done using statistical models where data from past time lags are used to forecast future value. The most notable technique is the Autoregressive Integrated Moving Average (ARIMA). The Recurrent Neural Network (RNN) is the neural network that models the dependence between sequences of inputs. The Long Short-Term Memory neural network (LSTM) is a version that can hold long-term information from data. Unlike the RNN, it has memory cells and gates that allow the model to remove irrelevant information from the last time step and remember important information from the current time step. This model demonstrated performance in various sequence problems such as translation, speech analysis, and voice recognition. We select the LSTM because it can model the temporal nature of time series data. Moreover, it provides a good benchmark for us to experiment and solve the multicollinearity aspect of algorithmic trading. The authors of [11] compared the performance of the ARIMA with the LSTM. They reported that the LSTM has an 85% improvement on prediction over the ARIMA in stock market indexes.

With sophisticated neural networks, technical indicators have been used to enhance algorithmic trading profitability, which can be seen in the case of Bursa Malaysia. It proved to outperform the return from the standard buy-and-hold strategy with the Kuala Lumpur Composite Index (KLCI) as a proxy for the stock market [12]. In recent years, [13] has proved that the deep LSTM can predict short term trend in foreign exchange rates and leads to increased profits and reduced drawdowns.

2.2. Solving the Multicollinearity Problem

Generally, researchers try to mitigate the effects of multicollinearity by using feature selection techniques to obtain a more reliable parameter estimate [14]. These are standard heuristic algorithms that rely on indicators. However, caution must be taken on compromising the theoretical model to reduce multicollinearity. There are three ways to accomplish this: the filter, wrapper, and embedded methods. A filter method evaluates variables based on specific selected measures. Examples of such properties are information gain and the chi-square test. However, the filter methods evaluate each feature individually and do not consider the interaction. Therefore, it is not very good in high multicollinearity data.

Meanwhile, a wrapper method evaluates the quality of a subset based on its learning algorithm. One of the earliest methods is stepwise regression. There are two basic ways of performing it: forward selection and backward elimination. The forward selection method starts with an empty model. Then, it adds variables one at a time. The backward elimination method starts with the full model with all available variables and drops them one by one. In each stage, they select the variable with the highest decrease in the residual sum of squares for forward selection or the lowest increase in the residual sum of squares for backward elimination. However, there are some drawbacks to stepwise regression. According to [15], it does not necessarily yield the best model due to the order that these variables are added. It is especially true in the presence of multicollinearity. Furthermore, it is also more costly than the filter method in computation and time. An exhaustive feature subset search may achieve the best result, but it is not feasible.

Embedded methods are learning algorithms that have built-in feature selection properties. They have the advantage of considering the relationship between features when training is fast. Such approaches use biased and shrunken estimators in exchange for lower variance and thus reduce overfitting. The advantage is that the theoretical model is not compromised because of the dropping of variables. Its disadvantage is that the estimators are now biased. The most known method is the ridge regression developed by [16]. This method adds a penalty term, the squared magnitude of the coefficient β, to the loss function. The general equation of ridge regression is as follows

$$\sum_{i=1}^{n} \left(Y_i - \sum_{j=1}^{p} X_{ij}\beta_j\right)^2 + \lambda \sum_{j=1}^{p} \beta_j^{\,2} \qquad (1)$$

Embedded feature selection aims to improve the efficiency in parameter estimation in the presence of multicollinearity. However, it comes with a bias-variance trade-off. Researchers can select the methods based on their purposes, such as the grouping effect or parsimony. For example, ref. [17] introduced a strictly concave penalty function called modified log penalty. It is contrary to the strictly convex penalty of the Elastic net. It is aimed to achieve a parsimonious model even under the effects of multicollinearity. Methods like the Elastic net tend to focus on the grouping effect, which means that the collinear variables are included together. It can require plenty of knowledge to know which one works better on the problem. Some methods work better in high or low dimensionality and degrees of multicollinearity. Besides that, some methods are for linear regression, and modifications need to be made for other functional form predictions or classification problems.

Ref. [18] proposed a hybrid method that combined factor analysis and an artificial neural network to combat multicollinearity. The ANN is not able to do variable selection; therefore, the PCA is used to extract components. The ANN is then applied on the components. This method is named FA-ANN (factor analysis–artificial neural network). It is compared with regression analysis and genetic programming. FA-ANN has the best accuracy among them. The advantage of FA-ANN and genetic programming is that they are not based on any statistical assumptions, and so they are more reliable and trustworthy.

The work in [19] provided reasons for why a machine learning algorithm might be better. They have no requirement for assumptions about the function, can uncover complex patterns, and dynamically learn changing relationships. Regularization and penalty mechanisms can also solve multicollinearity in machine learning models, for example, the Regularized OS-ELM algorithm [20] and the Least Squares Incremental ELM algorithm [21]. However, these mechanisms increase the computational complexity. For this reason, [22] proposed a method called the Kalman Learning Machine (KLM). It is an Extreme Learning Machine (ELM) that uses a Kalman filter to update the output weights of a Single Layer Feedforward Network (SLFN). The Kalman filter is an equation that can efficiently estimate the state of a process that minimizes the mean squared error. The state does not get updated in the learning stage as with the concept of the ELM. The resulting model has shown to outperform basic machine learning models in prediction error (RMSE) and computing time. However, it requires manual optimization by humans.

The feature selection methods aim to reduce the number of variables to the most relevant ones, which may reduce the information gained from having more data. Furthermore, modern optimization methods depend on subjectively determined indicators of relevance and similarity, which can be seen from [23], suggesting other measures of multicollinearity for future research. It is, therefore, difficult to suggest which method is better without directly comparing performance on the same dataset. Moreover, it is hard and computationally expensive to find a global best subset without an exhaustive search.

The findings in the literature show that feature selection drops the variable and reduces information gain. The multicollinearity measures to optimize are subjective and the global minimum is not guaranteed. Furthermore, each method has inconsistent performance depending on the data and are not applicable in every problem. In addition, a hybrid or ensemble method has promising performance and may improve financial forecasting. It combines the qualities of the filter, wrapper, and embedded methods. This paper suggests that the relevancy and redundancy concepts from feature selection can be adopted. Both relevance and redundancies can be learned along with the construction of the attention layer and the predictive model. The literature review also shows that machine learning algorithms are better than the simple OLS estimator in fitting data with multicollinearity. It does not need to have information on the relationship of the data or the distribution. This motivated us to use the LSTM as the predictive model.

3. Methodology

3.1. Data Collection

The data used in this research is the daily foreign exchange rate. We selected eight different pairs of exchange rates. These are EUR/GBP, EUR/JPY, EUR/USD, GBP/USD, NZD/USD, USD/CAD, USD/CHF, and USD/JPY. They are the foreign exchange pairing with the highest trading volume and trading data. The timeframe is the 5 years from 1 January 2015 to 31 December 2020. We collected the exchange rate at an hourly interval. The predictor variables are technical analysis indicators that are highly correlated. Neural networks using technical indicators have shown predictive capabilities comparable to a buy-and-hold strategy [24]. Technical analysis assumes that prices reflect all relevant information and predicts price movements based on asset price history and volume trends [25]. The following nine technical indicators are used as features for the proposed LSTM model.

3.1.1. Relative Strength Index (*RSI*)

The *RSI* is a momentum indicator used to measure the velocity and magnitude of a price movement. The *RSI* is between 0 and 100. When the index is above 70, it is considered overbought; it is considered oversold below 30 [26]. The average gain and average loss over a certain period are the relative strengths. The indicator is computed as follows

$$RSI = 100 - \frac{100}{1 + \frac{Average\ Gain}{Average\ Loss}} \tag{2}$$

3.1.2. Moving Average Convergence and Divergence (*MACD*)

The *MACD* indicator comprises two lines, the *MACD* line and the signal line. The *MACD* line is the difference between two *EMA*s of different periods. The signal line is an *EMA* of the *MACD*. The interpretation is that if the *MACD* line intersects the signal line downward, it indicates a downtrend, and vice versa if it crosses upward.

$$MACD\ Line : (12\ Days\ EMA - 26\ days\ EMA) \tag{3}$$

$$Signal\ Line : 9\ days\ EMA\ of\ MACD\ line \tag{4}$$

where L is the lowest price over period n and H is the highest price over period n. The trading signal is when $\%K$ and $\%D$ crosses.

3.1.3. Parabolic Stop and Reverse (*SAR*)

SAR is developed to identify trends in prices. Generally, the *SAR* below price is bullish and the *SAR* above is bearish. The formula is as below

$$SAR_{n+1} = SAR_n + \alpha(EP - SAR_{n)} \tag{5}$$

where EP, the extreme point, is the highest or lowest value achieved by an uptrend or downtrend in a period n, and α represents the acceleration factor and is initiated with 0.02. It explains the sensitivity of the *SAR*. Each time a new *EP* is recorded, the factor increases by 0.02, and thus the *SAR* will converge to the price faster.

3.1.4. Simple Moving Average (*SMA*)

The *SMA* is an unweighted moving average of the closing price of the previous n days. The number n can be selected depending on the period of trend desired, for example, short term, medium-term, and long-term trends. The *SMA* line can be used as the support or resistance level of the stock or used in conjunction with the *SMA* line of different periods to determine if it is on an uptrend or downtrend. For example, the stock price above the *SMA* indicated an uptrend and, therefore, a buy signal.

$$SMA = \frac{A_1 + A_2 + A_3 + A_4}{n} \tag{6}$$

where A_n is the price at period n and n is the total number of periods.

3.1.5. Cumulative Moving Average (*CMA*)

CMA is a moving average where the current average is cumulative of all the data until the current data point

$$CMA_4 = \frac{A_1 + A_2 + A_3 + A_4}{4} \tag{7}$$

where A_n is the price at period n.

3.1.6. Exponential Moving Average (EMA)

The EMA is a type of weighted moving average. The weighting of the latest data is the highest while the weight of the older data decreases exponentially. It is calculated with the formula below

$$S_t = \alpha \times Y_t + (1 + \alpha) \times S_{t-1} \tag{8}$$

where α is the parameter for the degree of decrease in weight and Y_t is the observation at time t.

3.1.7. Stochastic Oscillator

A stochastic oscillator is a momentum indicator that uses the price range over a period. The current price is a percentage of the recent highest and lowest price. The intuition is that prices tend to approach the extremes of the range before turning. The following is the calculation of the oscillator.

$$\%K = 100\frac{closing\ price - L}{H - L} \tag{9}$$

$$\%D = 3\ period\ moving\ average\ of\ \%K \tag{10}$$

3.1.8. William %*R*

The $\%R$ is an oscillator between -100 and 0, showing whether the stock price is trading near the highs or lows of the recent period. A value of -100 means that the closing price is near the low of the past n days.

$$\%R = \frac{(high) - close}{-min(low)} * -100 \tag{11}$$

3.1.9. Bollinger Band

The Bollinger Band is an indicator that uses highness and lowness relative to the price. It consists of an upper and a lower band. The price approaching near either band indicates a potential reversal in trend.

$$Bollinger\ High = MA_n + 2 * \sigma_n \tag{12}$$

$$Bollinger\ Low = MA_n + 2 * \sigma_n \tag{13}$$

where MA is the simple moving average, σ is the standard deviation, and n is the number of days in the smoothing period.

3.2. Multicollinearity Analysis

We performed a multicollinearity diagnosis on each variable to ascertain the level of dependencies between the variables. We use the Variation Inflation Factor (*VIF*) for this purpose.

$$VIF_k = \frac{1}{1 - R_k^2} \tag{14}$$

To compute the *VIF* of the *k*-th variable, take the *k*-th variable as the explained variable of all the remaining variables and estimate the regression coefficient. Next, take the coefficient of the determination (R^2) of the regression and calculate with the formula. There is no formal value of the VIF to determine the presence of multicollinearity, but a value of 10 often indicates multicollinearity [27]. Therefore, a *VIF* value higher than 10 indicates the severe presence of multicollinearity. The result for each variable for the EUR/GBP dataset is shown in Table 1. The diagnosis shows that the datasets are highly multicollinear.

Table 1. Variation Inflation Factor Analysis for EUR/GBP dataset.

Variables	***VIF***
open	20,637
high	17,431
low	20,848
close	24,950
RSI	7
MACD	21
SAR	877
SMA 5	13,692
SMA 10	3599
SMA 20	243,345
CMA	4
EMA	67,907
%*K*	32
%*D*	13
%*R*	19
Bollinger High	61,274
Bollinger Low	62,111

3.3. Data Generation

The samples are constructed with a rolling window mechanism. Figure 1 shows an example of the mechanism with a time series of 10 time steps. With a window size of 3 time steps, the first row of data is t = 1 to t = 3, the second row of data is t = 2 to t = 4, and the third row of data is t = 3 to t = 5. With this mechanism, eight rows of data will be computed. This research used a lag of 10 trading days (window size). The target label is a classification of profit or loss. The target label is based on the next two trading days with the following criteria. Label = 1 (significant profit) if the price reaches a predetermined take profit price level within two trading days and label = 0 (significant loss) if it reaches stop loss. If the price after two trading days is in between the two price levels, it will be labeled 3 (profit) if the trade still profits and 2 (loss) if it is a loss. This labeling methodology is shown in Figure 2.

3.4. Model Framework

The historical prices of foreign exchanges used in our paper are also sequential. The LSTM network can learn potential temporal information from the sequential data. The LSTM network has shown outstanding performance on many real-world applications that involve sequential data. Therefore, an LSTM model serves as a suitable baseline model for this study. The LSTM is a type of RNN. However, the RNN cannot perform well in long-sequence data due to exploding and vanishing gradients. Therefore, the LSTM model is created to solve this problem [28]. Unlike the RNN, it has memory gates that capture and retain information from long time lags while selectively removing stored information. Figure 3 displays the cell of an LSTM network. The top horizontal line is the cell state. The four neural net layers are gates that control what information to add or remove. At lag t, the first sigmoid (σ) layer decides what to forget. It takes the previous hidden state h_{t-1} and current input x_t to output a value between 0 and 1. Hidden state h_{t-1} is the encoded input of the previous time step, while x_t is the input of time step t, containing every feature. The sigmoid output value of zero means to forget completely, and one means to keep completely. The next sigmoid layer, called the input gate, decides which value to update. A next tanh layer creates new values to add to the cell state. A pointwise multiplication combines these two to update the cell state. Lastly, a sigmoid layer decides which part of the cell state to output.

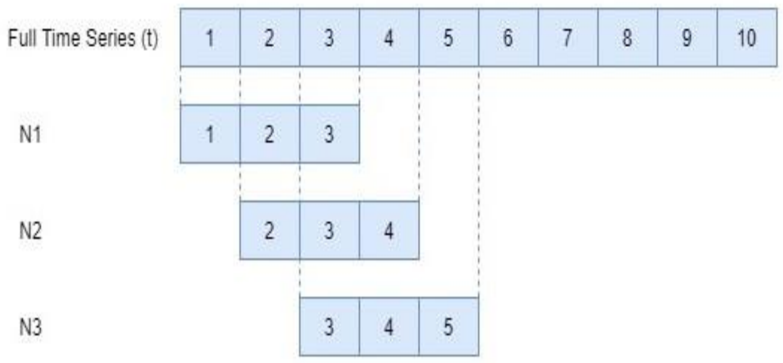

Figure 1. Illustration of the rolling window mechanism.

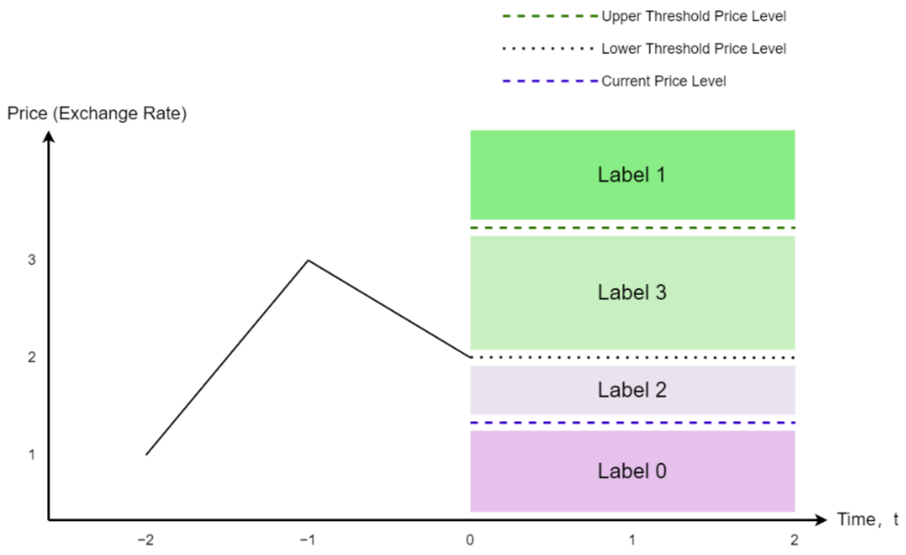

Figure 2. Illustration of dataset labeling methodology.

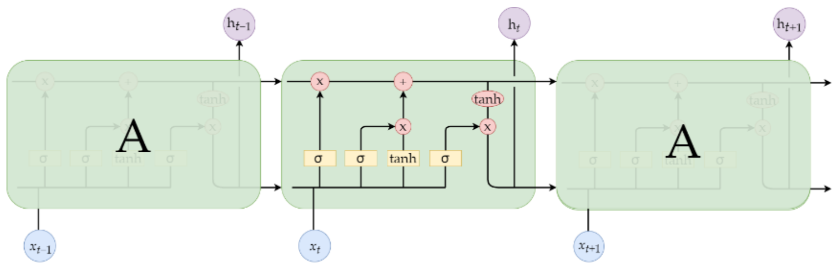

Figure 3. LSTM module diagram.

Our proposed model used the attention mechanism. It was developed for neural machine translation by [8]. It allows a model to search for segments of a source sentence that are most relevant to the target word. Furthermore, attention weighting aligns with the author's intuition. We use this mechanism to weigh the relevance of each feature to the target variable. For example, one Moving Average indicator might be more relevant to the target than another due to the different time frames used. The attention module consists of a linear layer, dropout layer, sigmoid activation, another linear layer, and a SoftMax layer. The resulting weight of the module is a learned measure of importance for each feature. Finally, the SoftMax function is applied to ensure that all the weightings sum to one.

The next part of our model is correlation embedding. In this section, we explain how the correlation embeddings are derived. We use this as a proxy for redundancy information between features. A correlation measures the strength of the relationship between the relative movements of two variables. The correlation is computed with the formula below

$$R_{ij} = \frac{C_{ij}}{\sqrt{C_{ii} * C_{jj}}} \tag{15}$$

where R_{ij} is the correlation coefficient of x_i and x_j, C_{ij} is the covariance matrix of x_i and x_j, C_{ii} is the variance, and C_{ij} is the variance of x_j. We calculate the correlation coefficient for each pair of features in our dataset. These calculations result in a correlation matrix. The values range from -1.0 to 1.0. A correlation of -1.0 indicates a perfect negative correlation, while 1.0 indicates a perfect positive correlation. For example, the Bollinger High and Bollinger Low are highly collinear due to being the upper and lower bands of price. The correlation matrix is then fed into a neural network layer with an output size similar to the input data. The output of this is known as the correlation embedding.

The general framework of the proposed model is illustrated in Figure 4. It is referred to as the Multicollinearity Reduction Model (MRM). The model receives an input in the shape of the batch, sequence length, input size, and it is denoted as X_t. This input data is fed into the input attention module, which generated a weight for each feature. This weighting was multiplied with the input data to get weighted input data. Next, we multiplied the weighted input and correlation before feeding it into an LSTM layer. This correlation embedding is denoted as *cr* in the diagram. The LSTM layer will have information on the relevance of each feature and the redundancy between the features. The intuition is that the model can predict using its learned attention and the correlation information of the features. This way, the model does not need to remove features to get good results.

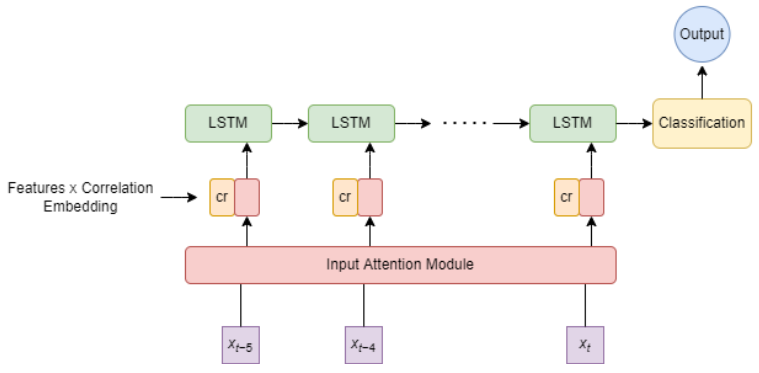

Figure 4. Proposed Multicollinearity Reduction Model Framework.

The classification network takes in the final output of the LSTM and passes through two more linear layers. The resulting output is the prediction of classes. The LSTM layer and the classification layer are what we will use as a baseline model. Figure 5 shows the overall methodology of our paper. The first step is gathering raw data, and it includes retrieving the historical prices of foreign exchange. The next step is preprocessing the data for analysis. It consists of generating samples and the target label, feature engineering, and normalization. Lastly, the data is fed into the predictive model for forecasting and comparison with the baseline results.

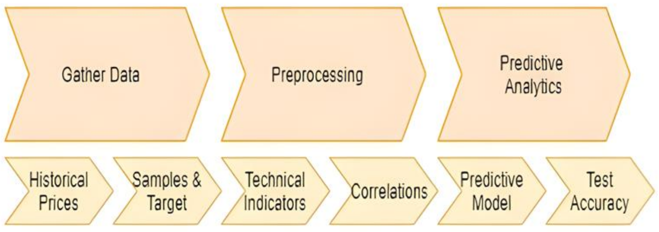

Figure 5. Overview of methodology for foreign exchange prediction based on technical indicators.

4. Performance Results

The architecture of our baseline algorithmic trading network contains three fully connected layers. We implemented all the algorithms in the Python programming language. The first layer is the LSTM recurrent layer. The input size is 17, and it contains 100 neurons in the hidden state. The second layer is a linear layer with 50 neurons. The final layer contains four neurons (significant loss, loss, profit, significant profit) for the model's output. It results in a total trainable parameter of 52,701. The backpropagation algorithm we use is the Adam optimizer [23].

The weights and biases of the LSTM layer were initialized with a uniform distribution from $-\sqrt{k}$ to $\sqrt{k}$, where k = 1/hidden size. The initial hidden state and cell state of the LSTM layer were initialized as zero. The weights and biases were similarly initiated with k = number of input features for the subsequent linear layers.

We configured the learning rates for each model independently to achieve the lowest error possible. The same learning rate was used for all layers. We configured the learning rate based on a simple heuristic. The learning rate was first initiated with 0.1. Then, it was divided by 10 until no further improvements in cross-entropy error were observed.

The effectiveness of the MRM was assessed by comparing it with the LSTM. The MRM architecture consists of the baseline LSTM, our proposed correlation embeddings, and the attention mechanism. Each model is trained over 100 epochs. The experiment was conducted with ten cycles, and the average was obtained. These steps were repeated for each of our eight datasets. The results are shown in Table 2. We first observe the cross-entropy loss of each model. The cross-entropy loss is useful in classification problems. It calculates the difference between the predicted and true probability distributions. The loss is described as follows

$$loss(x, class) = -x[class] + \log\left(\sum_j \exp\left(x[j]\right)\right) \tag{16}$$

where x is the input and class are the category index. The LSTM gave us a training loss of 0.843, while the MRM model achieved a training loss of 0.5088. The results proved that our proposed method is able to improve performance in high multicollinearity conditions.

Table 2. Accuracy, Returns, and Training Loss for each model.

Evaluation Metric	LSTM	MRM LSTM
Mean of Accuracy	45.43%	43.88%
Std of Accuracy	2.54%	2.62%
Mean of Returns (pips)	1476.14	2167.34
Std of Returns (pips)	452.01	525.88
Mean of Loss Function	0.8435	0.5088
Std of Loss Function	0.0241	0.0750

Next, we assess how our MRM model affects the trading performance. The metric considered is trading returns from the test set. The models are trained on the first 80% of data. The remaining 20% is used as the test set. There are approximately 7000 h of data. The profit and loss from the test set is the metric for comparison. The mean of trading return for the LSTM is 1476.14 pips, while the mean of trading return for the MRM is 2167.34 pips. That represented a 46.82% improvement in profit. The best result is achieved in EURGBP, with a trading improvement of 1232 pips. Meanwhile, only one out of the eight datasets did not have improved returns. Finally, we present a bar chart in Figure 6 to better visualize the performance.

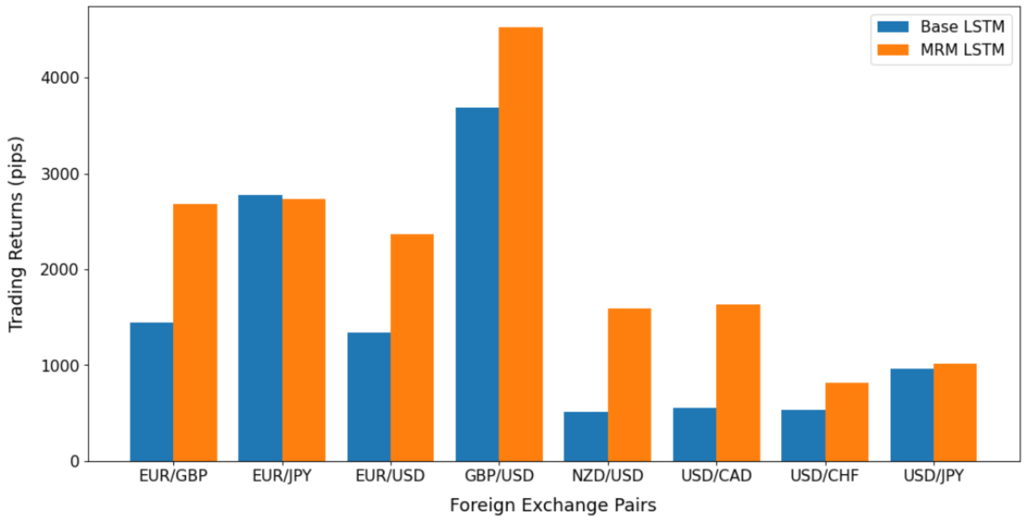

Figure 6. Comparison of Trading Returns for each of the Foreign Exchange Pairs.

While the returns showed significant improvement, that is not the case for accuracy (correct prediction of each label over total number prediction). The baseline model has an average accuracy of 45.43%. However, our proposed model achieved an accuracy of 43.88%. Figure 7 shows the accuracy of the proposed MRM model in a box plot. The mean and standard deviation of the accuracy are slightly higher than the LSTM model. The results suggest that the MRM model is better at identifying profitable trades than the LSTM model. As label 1 indicates a buy signal, it determines the trading returns of the model.

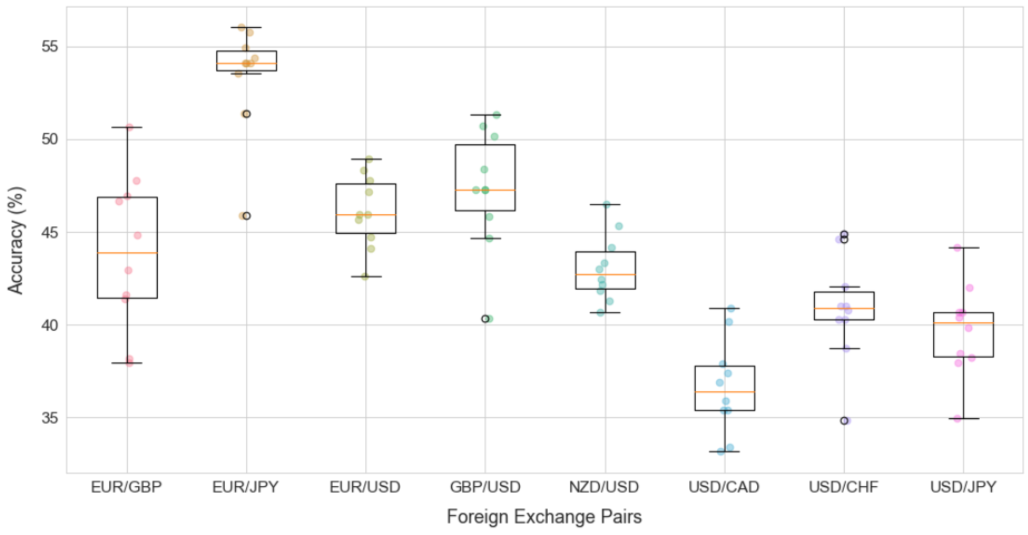

Figure 7. The MRM Model Prediction Accuracy for each of the Foreign Exchange Pairs.

The metrics examined proved that the MRM is an effective mechanism for high multicollinearity. Our proposed model can increase the profitability of algorithmic trading models. The LSTM is good at modeling temporal effects, but it does not consider the interaction between data. Our results show that neural networks with our extensions can learn the relevance and redundancy present in the financial data.

5. Conclusions

This paper explores the application of correlation embeddings and attention mechanisms to address the high multicollinearity phenomenon in algorithmic trading. We used the VIF and found that the multicollinearity problem is serious in a financial dataset. The introduced attention mechanism can learn each feature's relevance with the target variable to perform embedded feature selection. In addition, the correlation embeddings provide feature redundancy information for the model to make a prediction. The proposed MRM model can make a better prediction with these mechanisms. Our experiment revealed that accuracy does not necessarily translate to higher returns in an algorithmic trading simulation. Our model has higher profitability and returns with the same accuracy. Furthermore, this is achieved without a significant increase in training time. As such, we suggest that the MRM model be applied when dealing with high multicollinearity data.

The proposed method does not remove variables, making it better in prediction. Feature selection presents the risk of removing relevant features. Without an exhaustive search on a subset of features, it is difficult to arrive at the best model. Existing methods require plenty of knowledge and are based on a trial-and-error approach. Besides that, neural networks can uncover nonlinear relationships that most statistical approaches fail to perform.

Future work is of interest to expand the features beyond price-based technical indicators. For example, fundamental data and news data are data that are also used in algorithmic trading models. These data are also high in multicollinearity. The addition of features can assess the performance of our proposed MRM in higher-dimensional datasets. Furthermore, other measures besides correlation can be used experimentally as a proxy for redundancy in features.

Author Contributions: J.Y.-L.C. and S.M.H.L. investigated the ideas, reviewed the systems and methods, and wrote the manuscript; S.M.H.L. and K.T.B. provided the survey studies and methods; W.K.C. and S.W.P. conceived of the presented ideas and wrote the manuscript with support from J.Y.-L.C.; Z.-W.H. and J.-M.L. provided the suggestions on the experimental setup and provided the analytical results; J.Y.-L.C. and Y.-L.C. provided the suggestions on the research ideas, analytical results, wrote the manuscript, and provided funding supports. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the Fundamental Research Grant Scheme provided by the Ministry of Higher Education of Malaysia under grant number FRGS/1/2019/STG06/UTAR/03/1. It was also supported by Ministry of Science and Technology in Taiwan, grant MOST-109-2628-E-027-004–MY3 and MOST-110-2218-E-027-004, and was also supported by the Ministry of Education of Taiwan under Official Document No. 1100156712, entitled "The study of artificial intelligence and advanced semiconductor manufacturing for female STEM talent education and industry-university value-added cooperation pro-motion".

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Conflicts of Interest: No potential conflict of interest was reported by the authors.

References

1. Treleaven, P.; Galas, M.; Lalchand, V. Algorithmic trading review. *Commun. ACM* **2013**, *56*, 76–85. [CrossRef]
2. Daoud, J.I. Multicollinearity and Regression Analysis. *J. Phys. Conf. Ser.* **2017**, *949*, 012009. [CrossRef]
3. Bollinger, J. Using bollinger bands. *Stock. Commod.* **1992**, *10*, 47–51.
4. Khaire, U.M.; Dhanalakshmi, R. Stability of feature selection algorithm: A review. *J. King Saud Univ.-Comput. Inf. Sci.* **2019**, *34*, 1060–1073. [CrossRef]
5. Obite, C.P.; Olewuezi, N.P.; Ugwuanyim, G.U.; Bartholomew, D.C. Multicollinearity Effect in Regression Analysis: A Feed forward Artificial Neural Network Approach. *Asian J. Probab. Stat.* **2020**, *6*, 22–33. [CrossRef]

6. Wu, Y.-C.; Feng, J.-W. Development and Application of Artificial Neural Network. *Wirel. Pers. Commun.* **2018**, *102*, 1645–1656. [CrossRef]
7. Lucey, B.M.; Muckley, C. Robust global stock market interdependencies. *Int. Rev. Financ. Anal.* **2011**, *20*, 215–224. [CrossRef]
8. Bahdanau, D.; Cho, K.; Bengio, Y. Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv* **2014**, arXiv:1409.0473.
9. Yuan, H.; Zhou, H.; Cai, Z.; Zhang, S.; Wu, R. Dynamic Pyramid Attention Networks for multi-orientation object detection. *J. Internet Tech.* **2022**, *23*, 79–90.
10. Lee, M.-C. Research on the Feasibility of Applying GRU and Attention Mechanism Combined with Technical Indicators in Stock Trading Strategies. *Appl. Sci.* **2022**, *12*, 1007. [CrossRef]
11. Siami-Namini, S.; Namin, A.S. Forecasting economics and financial time series: ARIMA vs. LSTM. *arXiv* **2018**, arXiv:180306386.
12. M'ng, J.C.P.; Aziz, A.A. Using neural networks to enhance technical trading rule returns: A case with KLCI. *Athens J. Bus. Econ.* **2016**, *2*, 63–70. [CrossRef]
13. Rundo, F. Deep LSTM with Reinforcement Learning Layer for Financial Trend Prediction in FX High Frequency Trading Systems. *Appl. Sci.* **2019**, *9*, 4460. [CrossRef]
14. Katrutsa, A.; Strijov, V. Comprehensive study of feature selection methods to solve multicollinearity problem according to evaluation criteria. *Expert Syst. Appl.* **2017**, *76*, 1–11. [CrossRef]
15. Smith, G. Step away from stepwise. *J. Big Data* **2018**, *5*, 32. [CrossRef]
16. Horel, A. Applications of ridge analysis to regression problems. *Chem. Eng. Prog.* **1962**, *58*, 54–59.
17. Nguyen, V.C.; Ng, C.T. Variable selection under multicollinearity using modified log penalty. *J. Appl. Stat.* **2019**, *47*, 201–230. [CrossRef]
18. Garg, A.; Tai, K. Comparison of regression analysis, artificial neural network and genetic programming in handling the multicollinearity problem. In Proceedings of the 2012 International Conference on Modelling, Identifi-Cation and Control, IEEE, Wuhan, China, 24–26 June 2012; pp. 353–358.
19. Rasekhschaffe, K.C.; Jones, R.C. Machine Learning for Stock Selection. *Financ. Anal. J.* **2019**, *75*, 70–88. [CrossRef]
20. Huynh, H.; Won, Y. Regularized online sequential learning algorithm for single-hidden layer feedforward neural networks. *Pattern Recognit. Lett.* **2011**, *32*, 1930–1935. [CrossRef]
21. Guo, L.; Hao, J.-H.; Liu, M. An incremental extreme learning machine for online sequential learning problems. *Neurocomputing* **2014**, *128*, 50–58. [CrossRef]
22. Nóbrega, J.P.; Oliveira, A.L. A sequential learning method with Kalman filter and extreme learning machine for regression and time series forecasting. *Neurocomputing* **2019**, *337*, 235–250. [CrossRef]
23. Tamura, R.; Kobayashi, K.; Takano, Y.; Miyashiro, R.; Nakata, K.; Matsui, T. Best subset selection for eliminating multicollinearity. *J. Oper. Res. Soc. Jpn.* **2017**, *60*, 321–336. [CrossRef]
24. Sezer, O.B.; Ozbayoglu, A.M.; Dogdu, E. An artificial neural network-based stock trading system using tech-nical analysis and big data framework. In Proceedings of the Southeast Conference ACMSE, Kennesaw, GA, USA, 13–15 April 2017.
25. Nuti, G.; Mirghaemi, M.; Treleaven, P.; Yingsaeree, C. Algorithmic trading. *Computer* **2011**, *44*, 61–69. [CrossRef]
26. Krishnaveni, P.; Swarnam, S.; Prabakaran, V. An empirical study to analyse overbought and oversold periods of shares listed in CNX Bankex. *Int. J. Manag.* **2019**, *9*, 155–167.
27. Lavery, M.R.; Acharya, P.; Sivo, S.A.; Xu, L. Number of predictors and multicollinearity: What are their effects on error and bias in regression? *Commun. Stat.-Simul. Comput.* **2017**, *48*, 27–38. [CrossRef]
28. Althelaya, K.A.; El-Alfy, E.-S.M.; Mohammed, S. Evaluation of bidirectional LSTM for short-and long-term stock market prediction. In Proceedings of the 2018 9th International Conference on Information and Communication Systems, Irbid, Jordan, 3–5 April 2018; pp. 151–156.