A Novel Ensemble Learning Approach for Stock Market Prediction Based on Sentiment Analysis and the Sliding Window Method

Raymond Chiong, Zongwen Fan, Zhongyi Hu, and Sandeep Dhakal

***Abstract*— Financial news disclosures provide valuable information for traders and investors while making stock market investment decisions. Essential but challenging, the stock market prediction problem has attracted significant attention from both researchers and practitioners. Conventional machine learning models often fail to interpret the content of financial news due to the complexity and ambiguity of natural language used in the news. Inspired by the success of recurrent neural networks (RNNs) in sequential data processing, we propose an ensemble RNN approach (long short-term memory, gated recurrent unit, and SimpleRNN) to predict stock market movements. To avoid extracting tens of thousands of features using traditional natural language processing methods, we apply sentiment analysis and the sliding window method to extract only the most representative features. Our experimental results confirm the effectiveness of these two methods for feature extraction and show that the proposed ensemble approach is able to outperform other models under comparison.**

***Index Terms*— Ensemble learning, financial news, sentiment analysis, sliding window method, stock market prediction.**

I. Introduction

STOCK market prediction is a widely studied problem with highly complicated, nonlinear, and nonstationary movements [1]–[3]. Such movements make obtaining reasonable prediction accuracy very difficult [4]. Information from financial disclosures, which are strongly related to the movements of stock markets, underpins the investment decisions made by traders and investors [5]–[7]. For example, financial news provides significant information about the companies' background and real-world events relevant to stock market patterns [8]–[10] and, thus, represents valuable references while making decisions to buy or sell stocks [11], [12]. Even though humans can correctly interpret the content of financial news without any issues, the complexity and ambiguity of natural language [13], [14] used in the news make it a very challenging task for computer-based models. In a recent study, Seong and Nam [15] attempted to use the support vector machine (SVM) with multiple kernel learning based on financial news and different feature extraction and selection methods to predict the stock market movements. Their results demonstrated that feature extraction and selection play an important role in prediction performance.

The most common methods used for feature extraction include bag-of-words, noun phrases, named entities, term frequency-inverse document frequency (TF-IDF), n-gram, and embedding (e.g., Word2vec) [16], [17]. However, bag-of-words, noun phrases, and TF-IDF do not take the order of words, which is contextually very important, into consideration. While n-gram utilizes n words at a time [18], choosing the right value of n is difficult. Word2vec [19] can generate a word representation by converting words and phrases into a vector representation to capture semantic features [20], but it is not good at dealing with unknown words. A popular deep learning model, the recurrent neural network (RNN) can learn long sequences from contextual information [21] and is widely used in natural language processing to learn hierarchical representations of data [22]. Kraus and Feuerriegel [23] combined deep learning and transfer learning to predict the movements of the stock market with bag-of-words (tens of thousands of features were extracted from financial news) and demonstrated the effectiveness of deep learning for stock market prediction.

Time series, which uses past behavior to predict the future movements of the stock market [24], is a useful tool for studying the historical patterns of the stock market and is widely used in financial forecasting [25]–[31]. By looking back and analyzing n time steps using the sliding window method, the next time step can be predicted [32]. The historical pattern is, thus, captured to some extent [33]. Traditional approaches using time series, such as autoregressive integrated moving average [34], [35] and generalized autoregressive conditional heteroskedasticity [36] models, are effective only when the series is stationary. The stock market, however, is not stationary. Therefore, using only the sliding window method for feature extraction is not sufficient to build a prediction model that can provide good performance.

Textual sentiments in financial information can influence individual, firm, and market-level behavior and performance [37]. Therefore, sentiment extraction from financial

Manuscript received 2 February 2022; revised 18 April 2022; accepted 21 May 2022. This work was supported in part by the Australia-Germany Joint Research Cooperation Scheme under Grant G1600912, in part by the Natural Science Foundation of China under Grant 72171183, in part by the Natural Science Foundation of Hubei Province under Grant 2021CFB481, and in part by the Scientific Research Funds of Huaqiao University under Grant 21BS122. *(Corresponding author: Zongwen Fan.)*

Raymond Chiong and Sandeep Dhakal are with the School of Information and Physical Sciences, The University of Newcastle, Callaghan, NSW 2308, Australia (e-mail: raymond.chiong@newcastle.edu.au; sandeep.dhakal@uon.edu.au).

Zongwen Fan is with the School of Information and Physical Sciences, The University of Newcastle, Callaghan, NSW 2308, Australia, and also with the College of Computer Science and Technology, Huaqiao University, Xiamen 361021, China (e-mail: zongwen.fan@uon.edu.au).

Zhongyi Hu is with the School of Information Management, Wuhan University, Wuhan 430072, China (e-mail: zhongyi.hu@whu.edu.cn).

Digital Object Identifier 10.1109/TCSS.2022.3182375

disclosures has received increasing attention from researchers in recent years. For example, Bouktif *et al.* [38] extracted sentiments from tweets to predict the stock market. Li *et al.* [39] implemented a stock price prediction framework and found that the sentiments of financial disclosures have strong connections with the forecasting performance. Nassirtoussi *et al.* [40] discovered that the sentiments extracted from financial news significantly impact gains or losses in financial markets. Chiong *et al.* [41] applied a sentiment analysis-based machine learning approach for financial market prediction and obtained good prediction performance. Similarly, Lutz *et al.* [42] proposed a novel machine learning approach to predict the sentence-level polarity labels of financial news, and their model showed potential to assist investors in their decision-making.

Given this context and the discussion above, in this article, we propose an RNN-based ensemble model that uses sentiment analysis and the sliding window method for stock market prediction. Sentiment analysis is first applied to extract the two most informative features—polarity and subjectivity—from financial news. Next, the sliding window method with n time steps is utilized to capture the historical patterns of the stock market [43], [44]. With the extracted features, three widely used RNNs are incorporated to train the prediction models. These RNNs include the conventional RNN (SimpleRNN), gated recurrent unit (GRU), and long short-term memory (LSTM); they have been selected considering their promising performance in processing sequential data [45]. Ensemble learning [46], [47] is then applied to aggregate the results of the three RNN models, with their weights optimized by the particle swarm optimization (PSO) algorithm [41], to form an integrated weighted average output. This ensemble technique is an effective way of improving prediction performance. Six evaluation metrics, including the accuracy, balanced accuracy, precision, recall, F-measure (F1), and area under the curve (AUC), are used to evaluate our proposed ensemble model. In addition, the Wilcoxon rank-sum test [48] is applied to ascertain the statistical differences between the experimental results obtained by the models under comparison. Our results show that the proposed model is able to significantly outperform other models that it is compared with.

The main contributions of this work can be summarized as follows:

1) Sentiment analysis and the sliding window method are applied in combination during data preprocessing to reduce the number of features.
2) A novel ensemble learning mechanism, which integrates the SimpleRNN, GRU, and LSTM, is proposed for stock market prediction, and the weights of the base estimators are optimized by the PSO algorithm.
3) Comprehensive experimental results show that our proposed model outperforms other compared models in terms of the metrics considered; this confirms that the proposed data preprocessing and ensemble techniques are effective for improving stock market prediction.

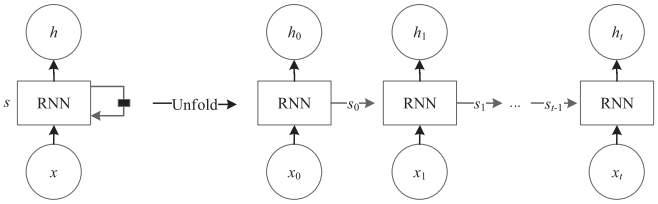


Fig. 1. Typical RNN and the unfolding structure with sequential computation. On the left-hand side, the notation of input is x, state is s, and output is h. The RNN unit can pass information from previous to the current state and then to the next state. Through this, it is able to memorize the previous information. On the right-hand side, the RNN unrolls over inputs $x_0, \ldots, x_t$, states $s_0, \ldots, s_{t-1}$, and outputs $h_0, \ldots, h_t$.

The remainder of this article is organized as follows. In Section II, the three RNNs (SimpleRNN, LSTM, and GRU) and the framework of our proposed ensemble RNN architecture are described. Experiments and analyses, including data preprocessing, the evaluation index, experimental results, and statistical analysis, are presented in Section III. Finally, conclusions are drawn in Section IV.

II. Methods

In this section, we first introduce three well-known RNNs capable of processing sequential structure data [49], namely the SimpleRNN, LSTM, and GRU. Following this, we describe our proposed architecture of the RNN ensemble.

A. SimpleRNN

RNNs are suitable for solving sequential problems [21], [50], thanks to their ability to memorize the historical information of sequential data. The biggest difference between them and traditional neural networks is that their data sequences can be dependent on each other. RNNs apply the same neural units to process every element in a sequence with the output depending on previous computations. A typical structure of SimpleRNN is given in Fig. 1.

As we can see from Fig. 1, the SimpleRNN is able to use historical information in arbitrarily long sequences and has been successfully used in text processing [51]. However, with the use of deep layers, backpropagated gradients either grow or shrink at each time step, resulting in the gradient vanishing or exploding problem [52], [53]; this limits their layers to looking back only a few steps in practice [49]. When moving from time step t to $t+1$, the SimpleRNN updates h_{t+1} from previous state s_t and current input x_{t+1} as follows:

$$h_{t+1} = \mathrm{RNN}(s_t, x_{t+1}). \tag{1}$$

B. LSTM

LSTM [54] was introduced to address the gradient vanishing or exploding problem of SimpleRNN. It processes long sequences by learning long-term dependencies of sentences [55], thus enabling them to memorize information for a longer period of time.

As we can see from Fig. 2, an LSTM unit has three types of gates (input gate, output gate, and forget gate) and one candidate vector to control the cell state, which interacts in a very special manner to process sequential information. This is opposed to a SimpleRNN that has only one neural network layer in an RNN unit. Such special structures can either forget information or save it to the cell state. The output of each gate ranges from zero to one, indicating the proportion of information that goes through the gate, with zero and one corresponding to "stop all information" and "let all information through," respectively. As state c_t passes forward, it can either remember or forget information manipulated by the three gates. Each gate has a neural network layer with its own sigmoid activation function, while the candidate vector has a tanh function as output. Specifically, the forget gate takes the output, h_{t-1}, from the previous information and returns a vector, f_t, which indicates how strongly the state c_t should be passed on to the next cell state, where zero means discarding and one means remembering all the information. The input gate takes h_{t-1} and x_t as inputs and returns a vector u_t, which is multiplied with the candidate values c'_t using elementwise multiplication (represented as $\odot$). Finally, the new cell c_t is translated into the output h_t using the output gate, which means that the information passes to the output. The new cell state and the new output are expressed in (6) and (7), respectively,

$$u_t = \sigma(\boldsymbol{W}_u \boldsymbol{X}_t + \boldsymbol{U}_u h_{t-1} + b_u) \tag{2}$$

$$f_t = \sigma(\boldsymbol{W}_f \boldsymbol{X}_t + \boldsymbol{U}_f h_{t-1} + b_f) \tag{3}$$

$$o_t = \sigma(\boldsymbol{W}_o \boldsymbol{X}_t + \boldsymbol{U}_o h_{t-1} + b_o) \tag{4}$$

where $\sigma(\cdot)$ is the sigmoid activation function, $\boldsymbol{W}$ and $\boldsymbol{U}$ are weight matrices between two layers trained by gradient descent, u_t is the input gate state, f_t is the forget gate state, c_t is the current state, and o_t is the output state

$$c'_t = \tanh(\boldsymbol{W}_c \boldsymbol{X}_t + \boldsymbol{U}_c h_{t-1} + b_c) \tag{5}$$

$$c_t = f_t \odot c_{t-1} \oplus u_t \odot c'_t \tag{6}$$

$$h_t = o_t \odot c_t \tag{7}$$

where $\odot$ is the elementwise multiplication between two states, c'_t is a candidate vector, and b_i is the bias for layer $i \in [u, f, o, c]$ in an LSTM unit.

C. GRU

Another method for solving the gradient vanishing or exploding problem is the GRU [56]. Similar to LSTM, the GRU also uses a gate structure but with only two gates: update gate and reset gate. The update and reset gates control the amount of information that should be passed to the output or discarded, respectively. Thus, by resetting or updating their memory content, gates can keep the information for a long period. At each time step, the GRU exposes its memory content and balances between the previous and new memory content using a leaky-integration unit. This makes the training time of GRU more efficient than LSTM. A single unit of GRU architecture is described in Fig. 3 to explain the mathematics behind the process of GRU.

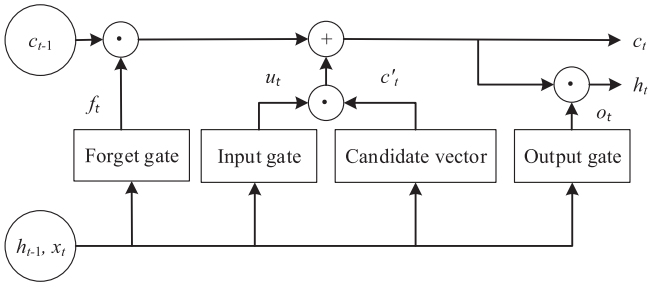


Fig. 2. LSTM architecture.

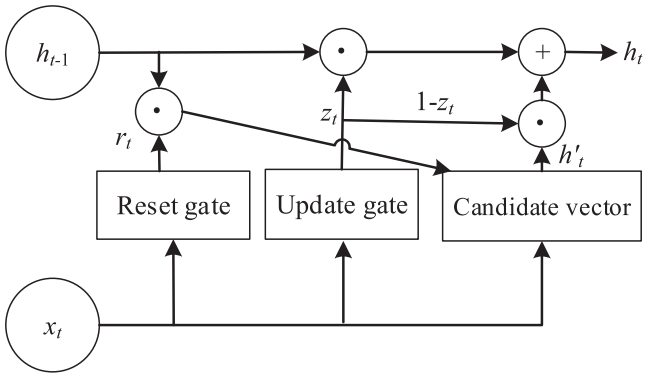


Fig. 3. GRU architecture.

As we can see from Fig. 3, given an input x_t, it is multiplied by the weight W_z when it goes through the network unit at the tth time step. The update gate z_t [see (8)] controls the amount of past information that can pass to the future time step, which is decided by a sigmoid function. The reset gate r_t [see (9)] controls the amount of past information that needs to be forgotten. h_t holds the information for the current unit and passes it to the next time step, which is expressed in (11). The operation of r_t is similar to the update gate but with different weights, which are W_r and U_r for X_t and h_{t-1}, respectively,

$$z_t = \sigma(\boldsymbol{W}_z \boldsymbol{X}_t + \boldsymbol{U}_z h_{t-1}) \tag{8}$$

$$r_t = \sigma(\boldsymbol{W}_r \boldsymbol{X}_t + \boldsymbol{U}_r h_{t-1}) \tag{9}$$

$$h'_t = \tanh(\boldsymbol{W} \boldsymbol{X}_t + r_t \boldsymbol{U} h_{t-1}) \tag{10}$$

$$h_t = z_t \odot h_{t-1} \oplus (1 - z_t) \odot h'_t \tag{11}$$

where h'_t holds the information at the tth time step using weights $\boldsymbol{W}$ and $\boldsymbol{U}$ with the tanh activation function.

D. RNN-Based Ensemble Learning

In most cases, the ensemble strategy is an effective way of obtaining better prediction performance than with individual models [47]. A flowchart of the proposed ensemble deep learning framework and its base models is depicted in Fig. 4. As we can see from the upper part of Fig. 4, the original dataset is first preprocessed by sentiment analysis and the sliding window method. Two kinds of features are then combined into the new dataset: one set of features is derived from the financial news and the other from stock market data. Next, three types of RNN models, namely the SimpleRNN, GRU, and LSTM, are designed to increase the model diversity. Finally, to improve

the prediction performance, these models are integrated to obtain the final results. The bottom part of Fig. 4 shows the structure of base classification models that we designed and used in our experiments together with their specific topologies.

The details of each base estimator are given as follows:

1) *SimpleRNN:* A SimpleRNN layer with 64 hidden neurons and a sigmoid activation function is utilized to process the sequential data, which yields 4608 parameters. The next layer—a dense layer with two hidden neurons and an exponential linear unit (elu) activation function—is a fully connected layer with 130 parameters. Then, the same fully connected layer is added but with one hidden neuron and the sigmoid activation function, resulting in three parameters. The final layer is a TimeDistributed layer with the sigmoid activation function, which allows it to apply a layer to every temporal slice of an input. This configuration yields two parameters. Thus, there are 4743 parameters in total, leading to approximately two training examples per parameter.
2) *GRU:* A GRU layer with 64 hidden neurons and the elu activation function is first utilized, which yields 13 824 parameters. As we can see from Figs. 1 and 3, the GRU layer has three times the parameters of SimpleRNN with the same number of hidden neurons. The following dense layer with the elu activation function and the TimeDistributed layer with the sigmoid activation function have the same parameters as the SimpleRNN model i.e., 130 and 3, respectively. Thus, there are 13 957 parameters in total, leading to approximately 0.7 training example per parameter.
3) *LSTM:* An LSTM layer with 64 hidden neurons and the elu activation function is used as the first layer, which yields 18 432 parameters. Comparing Figs. 1 and 2, we can see that LSTM, with the same number of hidden neurons as the SimpleRNN, has four times as many parameters. Next, another LSTM layer with 32 hidden neurons and a scaled elu (selu) activation function is applied, which results in 12 416 parameters. The next layer is a fully connected layer with 16 neurons and the elu activation function, which yields 528 parameters. As dropout is a powerful regularization technique to avoid overfitting [57], a dropout layer with a dropout value of 0.4 is applied by randomly selecting nodes to be dropped out with the probability of 40%. Finally, the last two layers are the same as the SimpleRNN model but with 17 and 2 parameters, respectively. Thus, there are 31 395 parameters in total, leading to approximately 0.3 training example per parameter.

Once trained, these base estimators are used to predict the testing samples. We then ensemble these results for the final output, where the ensemble weights, optimized by PSO, are applied to combine the individual prediction outputs into an integrated, weighted average output. Given a sample input x, the final output (ensemble step) of our proposed model $f(x)$ can be expressed as follows:

$$f(x) = \frac{\text{SimpleRNN}(x)w_1 + \text{GRU}(x)w_2 + \text{LSTM}(x)w_3}{w_1 + w_2 + w_3} \tag{12}$$

where w_1, w_2, and w_3 are the weights, optimized by PSO, for the SimpleRNN, GRU, and LSTM, respectively.

III. Experimental Setup and Results

In this section, we compare the performance of our proposed model with other state-of-art models based on a real-world financial disclosure dataset [58]. The financial news was first preprocessed with sentiment analysis, and the sliding window method was then applied to extract historical data. Six metrics, namely the accuracy, balanced accuracy, precision, recall, F1-score, and AUC, were used to measure the performance of all the models. Finally, the Wilcoxon rank-sum test was applied to examine the statistical differences between different models based on the balanced accuracy and F1-score results.

A. Data Preprocessing

The real-world financial disclosure dataset comprises 13 135 regulated German *ad hoc* announcements in English that can be downloaded from [58]. To be consistent with the data preprocessing done in [23], the disclosures of penny stocks with stock prices less than €5 and those published on nontrading days (e.g., weekends and holidays) were removed. This leaves us with 10 895 remaining samples. For the stock market prediction task, based on the sign of the corresponding return, positive disclosures were encoded as 1, while negative disclosures were encoded as 0. The statistical descriptions of the stock market data are provided in Table I.

Details of applying sentiment analysis and the sliding window method for feature extraction are given as follows:

1) *Sentiment Analysis:* It is a useful tool for extracting sentiments and opinions from the context [59], and prior studies have demonstrated its effectiveness for extracting sentiment features from financial news [41]. We applied a Python package TextBlob (http://textblob.readthedocs.io/en/dev/) to extract text sentiments from financial news (Message). Specifically, the two most informative features—polarity and subjectivity—were extracted from each financial news item, where each item contained approximately 169 words on average. The value of polarity ranges between -1 and 1, with -1 being very negative and 1 being very positive, whereas subjectivity ranges between 0 and 1, with 0 being very objective and 1 being very subjective.
2) *Sliding Window:* The sliding window method is able to store historical information for n time steps (window size) [60]. It is very effective for tracking the movement of the stock market by looking back n steps to predict future steps. In this study, an n-lagged window size was applied to store the historical pattern. However, the optimal lag order significantly influences the capture

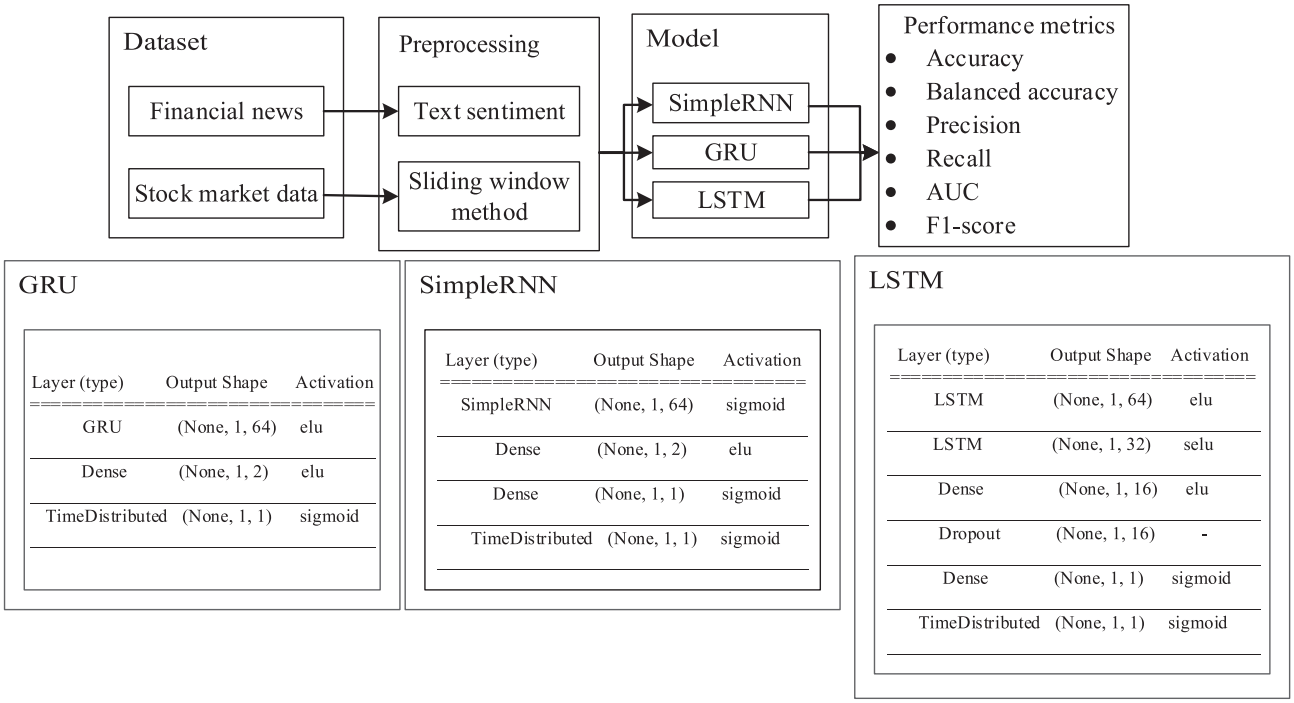


Fig. 4. Flowchart of the ensemble deep learning framework.

TABLE I
STATISTICAL DESCRIPTIONS OF THE STOCK MARKET DATA

Variable	Observation	Mean	Standard deviation
Abnormal return	10,895	0.699	6.923
Nominal return	10,895	0.778	6.876
Length of disclosures (in words)	10,895	168.801	117.397

of historical information [61]. Considering that there are five trading days in a week, we set the lag order $n = 5$. Specifically, each of the five-day lagged AbnormalReturnSign was added to input features. Hence, five-day lagged AbnormalReturnSign, and the two sentiment features, polarity and subjectivity, were used to predict AbnormalReturnSign.

Statistical properties of the dataset obtained after applying the above preprocessing techniques are provided in Table II with seven inputs and one output. It is worth noting that, although the first 80% of the samples were used as the training set (as in [23]) the number of training samples was different because of the use of the sliding window. As a result, our study has 8711 training samples compared to 8716 in [23]. The number of testing samples, however, remained the same.

B. Evaluation Metrics

As mentioned previously, the accuracy, balanced accuracy, precision, recall, F1-score, and AUC are used as performance measures in this study. Their calculations can be found in (13)–(17), respectively:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{13}$$

$$\text{Balanced accuracy} = \frac{1}{2}\left(\frac{\text{TP}}{\text{TP} + \text{FP}} + \frac{\text{TN}}{\text{TN} + \text{FN}}\right) \tag{14}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{15}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{16}$$

$$\text{F1-score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{17}$$

$$\text{AUC} = \frac{1}{2} + \frac{\text{TP}}{2(\text{TP} + \text{FN})} - \frac{\text{FP}}{2(\text{FP} + \text{TN})} \tag{18}$$

where TP is the number of true positives, TN is the number of true negatives, FP is the number of false positives, and FN is the number of false negatives in a confusion matrix, as shown in Table III.

C. Parameter Settings

As explained, the n in n-day lagged time series using the sliding window was set to 5. The models that are being compared in this study include the deep learning model proposed by Kraus and Feuerriegel [23], improved SVM optimized by PSO (PSO-SVM) [41], SimpleRNN, GRU, LSTM, and our proposed RNN-based ensemble model (denoted as EnsembleRNN). The parameter settings for the models are given as follows:

1) The deep learning model proposed by Kraus and Feuerriegel [23] was used for benchmarking purposes. They

TABLE II
STATISTICAL PROPERTIES OF THE STOCK MARKET DATASET AFTER SENTIMENT ANALYSIS AND TIME SERIES PREPROCESSING

Variables	Input/Output	Minimum	Maximum	Mean	Standard deviation
Day_t-5 AbnormalReturnSign	Input	0	1	0.5449	0.4980
Day_t-4 AbnormalReturnSign	Input	0	1	0.5450	0.4980
Day_t-3 AbnormalReturnSign	Input	0	1	0.5451	0.4980
Day_t-2 AbnormalReturnSign	Input	0	1	0.5451	0.4980
Day_t-1 AbnormalReturnSign	Input	0	1	0.5450	0.4980
Polarity	Input	-0.7	0.6	0.0767	0.0827
Subjectivity	Input	0	1	0.3572	0.1053
Day_t AbnormalReturnSign	Output	0	1	0.5450	0.4980

TABLE III
CONFUSION MATRIX FOR STOCK MARKET MOVEMENTS

		Predicted class	
		Positive (up)	Negative (down)
Actual class	Positive (up)	TP	FN
	Negative (down)	FP	TN

applied grid search with tenfold cross-validation; the model weights were initialized by the Xavier algorithm; the optimization function was Adam; the learning rate was tuned on the interval [0.0001, 0.01] with a step size of 0.0005; the size of word embedding was tuned on the interval [30, 100] with a step size of 10; the word embedding itself was initialized based on a continuous uniform distribution; and the size of each layer within their LSTM was the same as the dimension of the word embedding.

2) *PSO-SVM:* The penalty parameter, bandwidth of the Gaussian kernel, and epsilon-tube were optimized by the PSO algorithm, with its swarm size and maximum iteration set to 5 and 100, respectively.
3) *SimpleRNN:* Three hidden layers were used—one SimpleRNN layer with 64 neurons and the sigmoid activation function, and two dense layers with two neurons and one neuron each, and the elu and sigmoid activation functions, respectively. The number of neurons in TimeDistributed (output layer) was set to 1, and the sigmoid activation function was used for classification.
4) *GRU:* Two hidden layers were used, where the number of neurons in the GRU and dense layers were set to 64 and 2, respectively, and the elu activation function was used for both. The number of neurons in TimeDistributed was set to 1, and the sigmoid activation function was used for classification.
5) *LSTM:* Five hidden layers were used in the following order: two LSTM layers with 64 and 32 neurons each, a dense layer with 16 neurons, a dropout layer that can alleviate the overfitting problem, and a dense layer with 1 neuron. The corresponding activation functions for the five hidden layers were elu, selu, elu, no activation function for the dropout layer, and sigmoid, respectively. The number of neurons in TimeDistributed (output layer) was set to 1, and the sigmoid activation function was used for classification.
6) *EnsembleRNN:* Base estimators used were the SimpleRNN, GRU, and LSTM. Their corresponding weights used for the ensemble step were optimized by the PSO algorithm, where the swarm size and maximum iteration were set to 10 and 100, respectively; the weights were all randomly initialized within [0, 1]; and the upper and lower bounds for the weights were 0 and 1, respectively.

Some settings were common for the SimpleRNN, GRU, LSTM, and EnsembleRNN: the loss function, optimization function, and metrics were binary cross-entropy, Adam, and accuracy, respectively; and the minibatch size and epoch were set to 128 and 100, respectively. The structures of SimpleRNN, GRU, and LSTM can also be found in Fig. 4. All the methods being compared were trained and tested 30 times, and their average results were used as the final output.

D. Results and Discussion

To verify the effectiveness of sentiment analysis and the five-day lagged time series preprocessing (the sliding window method), we conducted experiments based on three different data preprocessing settings: 1) extracting features based on sentiment analysis only; 2) extracting features based on the sliding window method only; and 3) extracting features using both sentiment analysis and the sliding window method.

The results of our experiments conducted only with sentiment analysis preprocessing are shown in Table IV. The results show that the model based on deep learning and transfer learning [23] achieved the best performance in terms of accuracy (0.578) and balanced accuracy (0.583). This is because the model is trained based on thousands of features extracted from the traditional bag-of-words and TF-IDF preprocessing. In contrast, the PSO-SVM, SimpleRNN, GRU, LSTM, and our EnsembleRNN are trained based on two sentiment features (polarity and subjectivity), which is not enough to predict the movements of stock markets. Among all the prediction models, the GRU is the most efficient, followed by the SimpleRNN.

In the second set of experiments, which was conducted to verify the effectiveness of the sliding window method, a five-lagged window size was applied to store the historical pattern, given that there are five trading days in a week. As we can see from the results of these experiments, as shown in Table V, our proposed EnsembleRNN had the best accuracy, while the model by Kraus and Feuerriegel [23] had the best balanced accuracy. Comparing the results in Tables V with IV, we can see that the sliding window preprocessing method improved the performance of PSO-SVM, SimpleRNN, GRU, LSTM, and

TABLE IV
EXPERIMENTAL RESULTS BASED ON SENTIMENT ANALYSIS FOR ALL MODELS BEING COMPARED (THE BEST RESULTS ARE HIGHLIGHTED IN BOLD)

Algorithm	Accuracy	Balanced accuracy	AUC	Precision	Recall	F1-score	Computation time (s)
Kraus and Feuerriegel [23]	**0.5780**	**0.5830**	**0.5830**	**0.6099**	0.6304	0.6200	79200
PSO-SVM	0.5519	0.5327	0.5327	0.5473	0.8781	0.6740	5568
SimpleRNN	0.5396	0.5000	0.5000	0.5397	**0.9990**	**0.7008**	133
GRU	0.5385	0.5010	0.5010	0.5402	0.9738	0.6949	**126**
LSTM	0.5383	0.5001	0.5001	0.5397	0.9817	0.6965	137
EnsembleRNN	0.5395	0.5001	0.5001	0.5397	0.9966	0.7002	140

TABLE V
EXPERIMENTAL RESULTS BASED ON THE SLIDING WINDOW METHOD FOR ALL MODELS BEING COMPARED (THE BEST RESULTS ARE HIGHLIGHTED IN BOLD)

Algorithm	Accuracy	Balanced accuracy	AUC	Precision	Recall	F1-score	Computation time (s)
Kraus and Feuerriegel [23]	0.5780	**0.5830**	**0.5830**	**0.6099**	0.6304	0.6200	79200
PSO-SVM	0.5838	0.5818	0.5818	0.6034	0.6165	0.6099	4821
SimpleRNN	0.5855	0.5735	0.5735	0.5960	0.7247	0.6532	**41**
GRU	0.5790	0.5632	0.5632	0.5844	0.7625	**0.6614**	69
LSTM	0.5739	0.5571	0.5571	0.5795	0.7688	0.6605	59
EnsembleRNN	**0.5890**	0.5793	0.5793	0.6034	**0.7006**	0.6475	69

our EnsembleRNN in terms of all the metrics used. In terms of balanced accuracy, the improvements for the PSO-SVM, SimpleRNN, GRU, LSTM, and our EnsembleRNN are 9.22%, 14.70%, 12.42%, 11.40%, and 15.84%, respectively. This demonstrates that the historical trend of the stock market is helpful for predicting its future movements. In terms of computation time, the SimpleRNN was the most efficient, closely followed by LSTM based on the sliding window method for data preprocessing.

To make use of the information from different sources, we combined all the features extracted from the sentiment analysis and five-day lagged time series preprocessing. As reported in Table VI, our proposed EnsembleRNN performed the best in terms of accuracy, balanced accuracy, AUC, and precision, while LSTM had the best recall and F1-score, and the SimpleRNN needed the least computation time. The PSO-SVM, SimpleRNN, GRU, LSTM, and EnsemleRNN outperformed the model by Kraus and Feuerriegel [23] in terms of accuracy, recall, F1-score, and computation time. This is because TF-IDF in Kraus *et al.*'s model extracts tens of thousands of features, which makes the data very sparse, resulting in a larger number of features than training samples. In contrast, the other models extract only the seven most informative features based on sentiment analysis and the sliding window method. However, with the same sentiment analysis and five-day lagged preprocessing, the SimpleRNN and our ensemble model performed better than the PSO-SVM. RNN models can balance new and old information because of their ability to forget or remember past information, as required; this enables them to better address sequential problems [62]. Among the three RNN models, the SimpleRNN had the best performance in terms of accuracy, balanced accuracy, AUC, and precision; the LSTM had the best recall and F1-score; and the SimpleRNN had the least computation time. Overall, our proposed ensemble model had the best performance among all the models, except in terms of recall and F1-score. These results demonstrate the effectiveness of the proposed ensemble strategy to obtain better performance than individual models.

Furthermore, a comparison of the results in Table VI with Table V shows that data preprocessing based on sentiment analysis and the sliding window method could further improve the performance of SimpleRNN, GRU, LSTM, and our EnsembleRNN in terms of all the metrics used. For example, in terms of balanced accuracy, the improvements for the SimpleRNN, GRU, LSTM, and our EnsembleRNN are 1.27%, 0.55%, 1.78%, and 1.61%, respectively.

The results from Fig. 5 also prove that our proposed model has high prediction performance. In order to see the results of each experimental run more intuitively, comparison results based on 30 runs—in terms of accuracy, balanced accuracy, AUC, precision, recall, and F1-score—are shown in Fig. 6. As can be seen in the figure, the GRU and LSTM have wider fluctuations than the PSO-SVM, SimpleRNN, and our proposed EnsembleRNN in terms of accuracy, balanced accuracy, precision, recall, and F1-score. The proposed EnsembleRNN is the most stable in terms of all the metrics used.

E. Statistical Analysis

In this section, the Wilcoxon rank-sum test was applied to ascertain the statistical significance of the results obtained. By calculating the results of two vectors, the test obtained the two-sided p-value, which is used to check the null hypothesis if there is any significant difference between the comparison results. Here, we analyzed the balanced accuracy and F1-score results from 30 repeated runs. Tables VII and VIII show the statistical results of the Wilcoxon rank-sum test in terms of F1-score and balanced accuracy, respectively; p-values less than the significant threshold (0.05) are highlighted in bold, which means that their differences are significant.

As we can see from Table VII, all the p-values are less than the threshold 0.05 except the pair [GRU, LSTM], which means that the pairwise comparisons are significantly different,

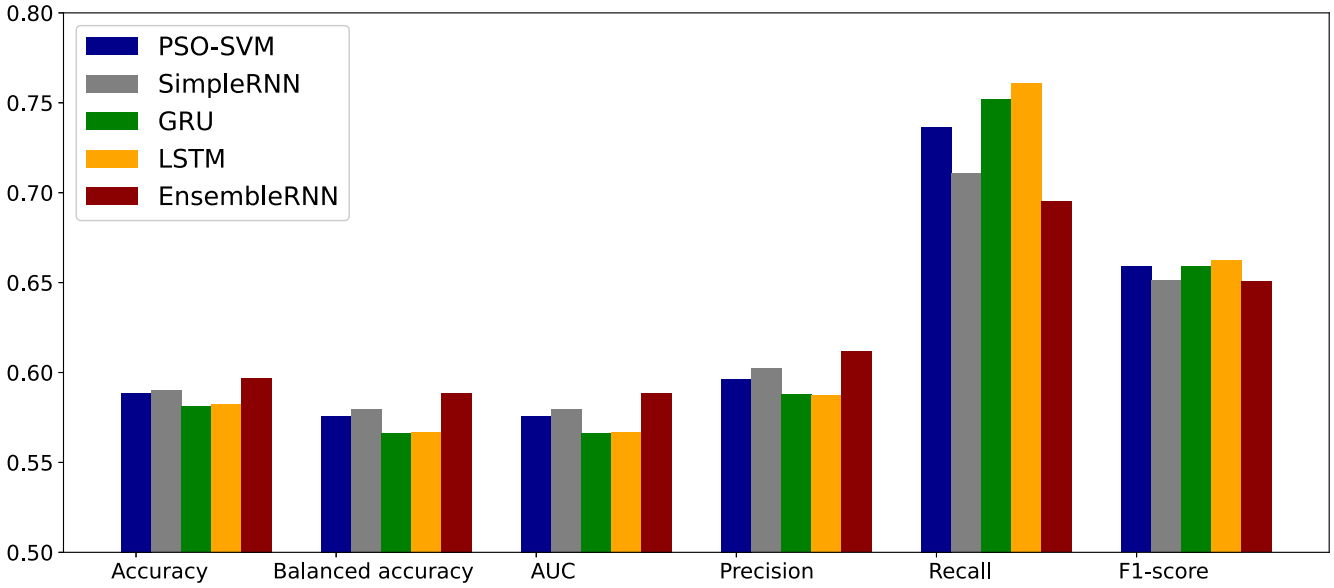


Fig. 5. Comparison of experimental results for stock market prediction in terms of accuracy, balanced accuracy, AUC, precision, recall, and F1-score.

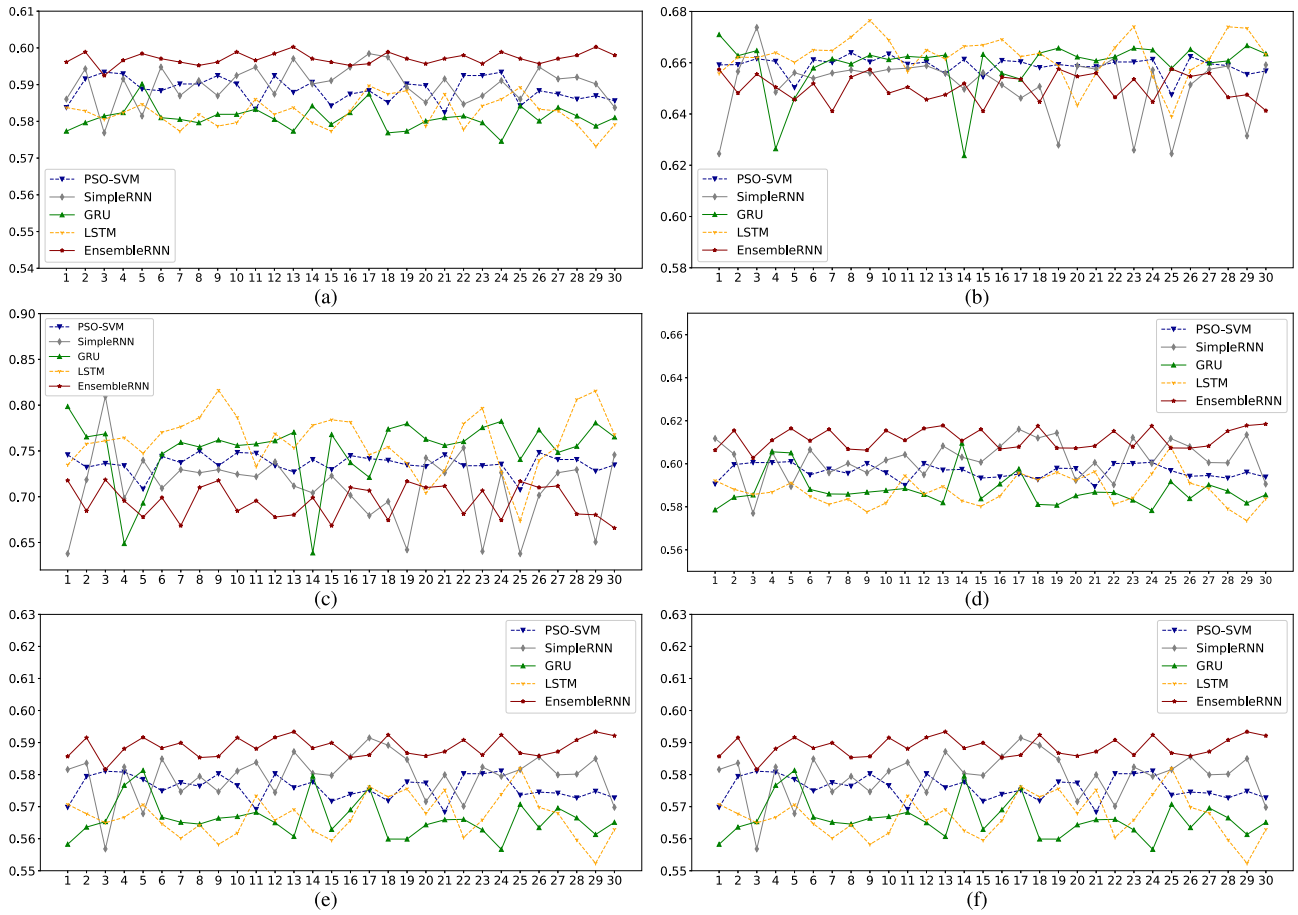


Fig. 6. Comparison results based on the experiments of 30 runs in terms of (a) accuracy, (b) balanced accuracy, (c) AUC, (d) precision, (e) recall, and (f) F1-score.

and the proposed EnsembleRNN significantly outperforms other models under comparison in terms of balanced accuracy. Similarly, we can see from Table VIII that most of the pairwise comparisons are significantly different except the pair [GRU, LSTM]. In addition, LSTM with sentiment analysis and the sliding window method for feature extraction significantly

TABLE VI

EXPERIMENTAL RESULTS BASED ON SENTIMENT ANALYSIS AND THE SLIDING WINDOW METHOD FOR ALL MODELS BEING COMPARED (THE BEST RESULTS ARE HIGHLIGHTED IN BOLD)

Algorithm	Accuracy	Balanced accuracy	AUC	Precision	Recall	F1-score	Computation time (s)
Kraus and Feuerriegel [23]	0.5780	0.5830	0.5830	0.6099	0.6304	0.6200	79200
PSO-SVM	0.5887	0.5759	0.5759	0.5964	0.7364	0.6590	9237
SimpleRNN	0.5907	0.5808	0.5808	0.6039	0.7055	0.6500	**130**
GRU	0.5810	0.5663	0.5663	0.5878	0.7520	0.6592	136
LSTM	0.5824	0.5670	0.5670	0.5876	**0.7610**	**0.6627**	147
EnsembleRNN	**0.5971**	**0.5886**	**0.5886**	**0.6116**	0.6951	0.6505	150

TABLE VII

STATISTICAL TEST RESULTS BASED ON THE WILCOXON RANK-SUM TEST BETWEEN EACH PAIR OF MODELS FOR BALANCED ACCURACY EVALUATION (p-VALUES LESS THAN 0.05 ARE HIGHLIGHTED IN BOLD)

Algorithm	PSO-SVM	SimpleRNN	LSTM	GRU	EnsembleRNN
Kraus and Feuerriegel [23]	$\mathbf{2.87\times10^{-11}}$	**0.027**	$\mathbf{2.87\times10^{-11}}$	$\mathbf{2.87\times10^{-11}}$	$\mathbf{5.32\times10^{-10}}$
PSO-SVM		$\mathbf{2.50\times10^{-3}}$	$\mathbf{3.66\times10^{-7}}$	$\mathbf{1.11\times10^{-7}}$	$\mathbf{2.87\times10^{-11}}$
SimpleRNN			$\mathbf{1.02\times10^{-7}}$	$\mathbf{2.49\times10^{-8}}$	$\mathbf{1.06\times10^{-8}}$
LSTM				0.57	$\mathbf{3.18\times10^{-11}}$
GRU					$\mathbf{2.87\times10^{-11}}$

TABLE VIII

STATISTICAL TEST RESULTS BASED ON THE WILCOXON RANK-SUM TEST BETWEEN EACH PAIR OF MODELS FOR F1-SCORE EVALUATION (p-VALUES LESS THAN 0.05 ARE HIGHLIGHTED IN BOLD)

Algorithm	PSO-SVM	SimpleRNN	LSTM	GRU	EnsembleRNN
Kraus and Feuerriegel [23]	$\mathbf{2.87\times10^{-11}}$	$\mathbf{2.87\times10^{-11}}$	$\mathbf{2.87\times10^{-11}}$	$\mathbf{2.87\times10^{-11}}$	$\mathbf{2.87\times10^{-11}}$
PSO-SVM		$\mathbf{3.83\times10^{-5}}$	$\mathbf{2.00\times10^{-3}}$	$\mathbf{7.45\times10^{-3}}$	$\mathbf{1.15\times10^{-8}}$
SimpleRNN			$\mathbf{1.38\times10^{-5}}$	$\mathbf{1.93\times10^{-5}}$	**0.029**
LSTM				0.18	$\mathbf{2.28\times10^{-7}}$
GRU					$\mathbf{2.28\times10^{-7}}$

outperforms the PSO-SVM in terms of F1-score. Among the SimpleRNN, GRU, and LSTM, the former outperforms the latter two in terms of balanced accuracy, but LSTM outperforms the SimpleRNN and GRU in terms of F1-score. To summarize, preprocessing with sentiment analysis and the sliding window method, and the ensemble strategy are able to improve the prediction performance.

IV. CONCLUSION

This article presented an RNN-based ensemble model for financial market prediction via news disclosures. Sentiment analysis and the sliding window method were applied to extract the most representative features from financial news and historical data. This greatly reduced the number of dimensions compared to traditional preprocessing strategies (e.g., bag-of-words and TF-IDF) that extract tens of thousands of features. With the extracted features, an ensemble approach of RNNs was applied for stock market prediction, and the results demonstrated that our ensemble model outperformed the other state-of-art models under comparison. It is worth noting that the methods implemented in our proposed model can be further explored in other application domains.

Since sentiments about financial news have a great impact on prediction accuracy, methods of training the sentiment analysis model to extract more accurate sentiments from financial news disclosures will be investigated in the future. Furthermore, the construction of more powerful RNN architectures with methods such as neural structure and parameter optimization will be studied. Finally, ways of integrating multisource information, such as financial news from different websites, for stock market prediction will be explored.

DECLARATION OF COMPETING INTEREST

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this article.

REFERENCES

[1] T. Fischer and C. Krauss, "Deep learning with long short-term memory networks for financial market predictions," *Eur. J. Oper. Res.*, vol. 270, no. 2, pp. 654–669, Dec. 2017.

[2] X. Wu, Q. Ye, H. Hong, and Y. Li, "Stock selection model based on machine learning with wisdom of experts and crowds," *IEEE Intell. Syst.*, vol. 35, no. 2, pp. 54–64, Mar. 2020.

[3] M. Jiang, L. Jia, Z. Chen, and W. Chen, "The two-stage machine learning ensemble models for stock price prediction by combining mode decomposition, extreme learning machine and improved harmony search algorithm," *Ann. Oper. Res.*, vol. 309, pp. 553–585, Jun. 2020.

[4] D. C. Furlaneto, L. S. Oliveira, D. Menotti, and G. D. C. Cavalcanti, "Bias effect on predicting market trends with EMD," *Expert Syst. Appl.*, vol. 82, pp. 19–26, Oct. 2017.

[5] I. E. Fisher, M. R. Garnsey, and M. E. Hughes, "Natural language processing in accounting, auditing and finance: A synthesis of the literature with a roadmap for future research," *Intell. Syst. Accounting, Finance Manage.*, vol. 23, no. 3, pp. 157–214, Jul. 2016.

[6] G. G. Creamer, "Network structure and market risk in the European equity market," *IEEE Syst. J.*, vol. 12, no. 2, pp. 1090–1098, Jun. 2018.

[7] S. Feuerriegel and J. Gordon, "Long-term stock index forecasting based on text mining of regulatory disclosures," *Decis. Support Syst.*, vol. 112, pp. 88–97, Aug. 2018.

[8] T. Geva and J. Zahavi, "Empirical evaluation of an automated intraday stock recommendation system incorporating both market data and textual news," *Decis. Support Syst.*, vol. 57, pp. 212–223, Jan. 2015.

[9] F. Z. Xing, E. Cambria, and R. E. Welsch, "Natural language based financial forecasting: A survey," *Artif. Intell. Rev.*, vol. 50, no. 1, pp. 49–73, Jun. 2018.

[10] K. Yuan, G. Liu, J. Wu, and H. Xiong, "Dancing with Trump in the stock market: A deep information echoing model," *ACM Trans. Intell. Syst. Technol.*, vol. 11, no. 5, pp. 1–22, Oct. 2020.

[11] K. Nam and N. Seong, "Financial news-based stock movement prediction using causality analysis of influence in the Korean stock market," *Decis. Support Syst.*, vol. 117, pp. 100–112, Feb. 2019.

[12] J. Nikkinen and J. Peltomäki, "Crash fears and stock market effects: Evidence from web searches and printed news articles," *J. Behav. Finance*, vol. 21, no. 2, pp. 117–127, 2019.

[13] S. Evert *et al.*, "Combining machine learning and semantic features in the classification of corporate disclosures," *J. Log., Lang. Inf.*, vol. 28, no. 2, pp. 309–330, 2019.

[14] E. Saquete, D. Tomás, P. Moreda, P. Martínez-Barco, and M. Palomar, "Fighting post-truth using natural language processing: A review and open challenges," *Expert Syst. Appl.*, vol. 141, Mar. 2020, Art. no. 112943.

[15] N. Seong and K. Nam, "Predicting stock movements based on financial news with segmentation," *Expert Syst. Appl.*, vol. 164, Feb. 2021, Art. no. 113988.

[16] R. Chiong, G. S. Budhi, S. Dhakal, and F. Chiong, "A textual-based featuring approach for depression detection using machine learning classifiers and social media texts," *Comput. Biol. Med.*, vol. 135, Aug. 2021, Art. no. 104499.

[17] S. F. A. Zaidi, F. M. Awan, M. Lee, H. Woo, and C.-G. Lee, "Applying convolutional neural networks with different word representation techniques to recommend bug fixers," *IEEE Access*, vol. 8, pp. 213729–213747, 2020.

[18] S. M. A. Islam, B. J. Heil, C. M. Kearney, and E. J. Baker, "Protein classification using modified *n-grams* and *skip-grams*," *Bioinformatics*, vol. 34, no. 9, pp. 1481–1487, May 2018.

[19] S. Yilmaz and S. Toklu, "A deep learning analysis on question classification task using Word2vec representations," *Neural Comput. Appl.*, vol. 32, pp. 2909–2928, Jan. 2020.

[20] J. Lilleberg, Y. Zhu, and Y. Zhang, "Support vector machines and Word2vec for text classification with semantic features," in *Proc. IEEE 14th Int. Conf. Cognit. Informat. Cognit. Comput. (ICCI*CC)*, Jul. 2015, pp. 136–140.

[21] Y. Qiu, H.-Y. Yang, S. Lu, and W. Chen, "A novel hybrid model based on recurrent neural networks for stock market timing," *Soft Comput.*, vol. 24, no. 20, pp. 15273–15290, 2020.

[22] T. Young, D. Hazarika, S. Poria, and E. Cambria, "Recent trends in deep learning based natural language processing," *IEEE Comput. Intell. Mag.*, vol. 13, no. 3, pp. 55–75, Aug. 2018.

[23] M. Kraus and S. Feuerriegel, "Decision support from financial disclosures with deep neural networks and transfer learning," *Decis. Support Syst.*, vol. 104, pp. 38–48, Dec. 2017.

[24] A. Atkins, M. Niranjan, and E. Gerding, "Financial news predicts stock market volatility better than close price," *J. Finance Data Sci.*, vol. 4, no. 2, pp. 120–137, Jun. 2018.

[25] Y. Liu, Y. Chen, S. Wu, G. Peng, and B. Lv, "Composite leading search index: A preprocessing method of internet search data for stock trends prediction," *Ann. Oper. Res.*, vol. 234, no. 1, pp. 77–94, Nov. 2015.

[26] R. Ren, D. D. Wu, and T. Liu, "Forecasting stock market movement direction using sentiment analysis and support vector machine," *IEEE Syst. J.*, vol. 13, no. 1, pp. 760–770, Mar. 2018.

[27] R. Efendi, N. Arbaiy, and M. M. Deris, "A new procedure in stock market forecasting based on fuzzy random auto-regression time series model," *Inf. Sci.*, vol. 441, pp. 113–132, May 2018.

[28] J. Cao, Z. Li, and J. Li, "Financial time series forecasting model based on CEEMDAN and LSTM," *Phys. A, Stat. Mech. Appl.*, vol. 519, pp. 127–139, Apr. 2019.

[29] M. Zhang, X. Jiang, Z. Fang, Y. Zeng, and K. Xu, "High-order hidden Markov model for trend prediction in financial time series," *Phys. A, Stat. Mech. Appl.*, vol. 517, pp. 1–12, Mar. 2019.

[30] F. Zhou, H.-M. Zhou, Z. Yang, and L. Yang, "EMD2FNN: A strategy combining empirical mode decomposition and factorization machine based neural network for stock market trend prediction," *Expert Syst. Appl.*, vol. 115, pp. 136–151, Jan. 2019.

[31] C. Lohrmann and P. Luukka, "Classification of intraday S&P500 returns with a random forest," *Int. J. Forecasting*, vol. 35, no. 1, pp. 390–407, Jan. 2019.

[32] G. Zhiqiang, W. Huaiqing, and L. Quan, "Financial time series forecasting using LPP and SVM optimized by PSO," *Soft Comput.*, vol. 17, no. 5, pp. 805–818, May 2013.

[33] O. B. Sezer, M. U. Gudelek, and A. M. Ozbayoglu, "Financial time series forecasting with deep learning: A systematic literature review: 2005–2019," *Appl. Soft Comput.*, vol. 90, May 2020, Art. no. 106181.

[34] F. Kaytez, "A hybrid approach based on autoregressive integrated moving average and least-square support vector machine for long-term forecasting of net electricity consumption," *Energy*, vol. 197, Apr. 2020, Art. no. 117200.

[35] M. Khedmati, F. Seifi, and M. J. Azizi, "Time series forecasting of bitcoin price based on autoregressive integrated moving average and machine learning approaches," *Int. J. Eng.*, vol. 33, no. 7, pp. 1293–1303, 2020.

[36] A. Aue, L. Horváth, and D. F. Pellatt, "Functional generalized autoregressive conditional heteroskedasticity," *J. Time Ser. Anal.*, vol. 38, no. 1, pp. 3–21, Jan. 2017.

[37] C. Kearney and S. Liu, "Textual sentiment in finance: A survey of methods and models," *Int. Rev. Financial Anal.*, vol. 33, pp. 171–185, May 2014.

[38] S. Bouktif, A. Fiaz, and M. Awad, "Augmented textual features-based stock market prediction," *IEEE Access*, vol. 8, pp. 40269–40282, 2020.

[39] X. Li, H. Xie, L. Chen, J. Wang, and X. Deng, "News impact on stock price return via sentiment analysis," *Knowl.-Based Syst.*, vol. 69, pp. 14–23, Oct. 2014.

[40] A. K. Nassirtoussi, S. Aghabozorgi, T. Y. Wah, and D. C. L. Ngo, "Text mining for market prediction: A systematic review," *Expert Syst. Appl.*, vol. 41, no. 16, pp. 7653–7670, 2014.

[41] R. Chiong, Z. Fan, Z. Hu, M. T. P. Adam, B. Lutz, and D. Neumann, "A sentiment analysis-based machine learning approach for financial market prediction via news disclosures," in *Proc. Genetic Evol. Comput. Conf. Companion*. New York, NY, USA: ACM Press, Jul. 2018, pp. 278–279.

[42] B. Lutz, N. Pröllochs, and D. Neumann, "Predicting sentence-level polarity labels of financial news using abnormal stock returns," *Expert Syst. Appl.*, vol. 148, Jun. 2020, Art. no. 113223.

[43] W. Chen, C. K. Yeo, C. T. Lau, and B. S. Lee, "Leveraging social media news to predict stock index movement using RNN-boost," *Data Knowl. Eng.*, vol. 118, pp. 14–24, Nov. 2018.

[44] W. Long, Z. Lu, and L. Cui, "Deep learning-based feature engineering for stock price movement prediction," *Knowl.-Based Syst.*, vol. 164, pp. 163–173, Jan. 2019.

[45] X.-Y. Zhang, F. Yin, Y.-M. Zhang, C.-L. Liu, and Y. Bengio, "Drawing and recognizing Chinese characters with recurrent neural network," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 849–862, Jun. 2017.

[46] G. Seni and J. F. Elder, "Ensemble methods in data mining: Improving accuracy through combining predictions," *Synth. Lectures Data Mining Knowl. Discovery*, vol. 2, no. 1, pp. 1–126, 2010.

[47] B. Weng, L. Lu, X. Wang, F. M. Megahed, and W. Martinez, "Predicting short-term stock prices using ensemble methods and online data sources," *Expert Syst. Appl.*, vol. 112, pp. 258–273, Dec. 2018.

[48] B. Derrick and P. White, "Comparing two samples from an individual Likert question," *Int. J. Math. Statist.*, vol. 18, no. 3, pp. 1–13, 2017.

[49] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org

[50] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436, 2015.

[51] A. Sherstinsky, "Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network," *Phys. D, Nonlinear Phenomena*, vol. 404, Mar. 2020, Art. no. 132306.

[52] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Trans. Neural Netw.*, vol. 5, no. 2, pp. 157–166, Mar. 1994.

[53] L. Guo, N. Li, F. Jia, Y. Lei, and J. Lin, "A recurrent neural network based health indicator for remaining useful life prediction of bearings," *Neurocomputing*, vol. 240, pp. 98–109, May 2017.

[54] X. Yuan, L. Li, and Y. Wang, "Nonlinear dynamic soft sensor modeling with supervised long short-term memory network," *IEEE Trans. Ind. Informat.*, vol. 16, no. 5, pp. 3168–3176, May 2019.

[55] L. Troiano, E. M. Villa, and V. Loia, "Replicating a trading strategy by means of LSTM for financial industry applications," *IEEE Trans. Ind. Informat.*, vol. 14, no. 7, pp. 3226–3234, Jul. 2018.

[56] M. Afrasiabi, M. Mohammadi, M. Rastegar, and S. Afrasiabi, "Advanced deep learning approach for probabilistic wind speed forecasting," *IEEE Trans. Ind. Informat.*, vol. 17, no. 1, pp. 720–727, Jan. 2021.

[57] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jan. 2014.

[58] M. Kraus and S. Feuerriegel. (2017). *Financial Deep Learning*. [Online]. Available: https://github.com/MathiasKraus/FinancialDeepLearning

[59] B. Liu, *Sentiment Analysis: Mining Opinions, Sentiments, and Emotions*. Cambridge, U.K.: Cambridge Univ. Press, 2015.

[60] B. LeBaron, W. B. Arthur, and R. Palmer, "Time series properties of an artificial stock market," *J. Econ. Dyn. Control*, vol. 23, no. 9, pp. 1487–1516, 1999.

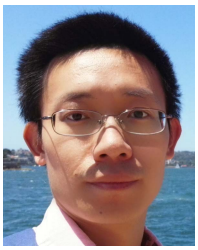
[61] H. Kim and J.-T. Lee, "On inferences about lag effects using lag models in air pollution time-series studies," *Environ. Res.*, vol. 171, pp. 134–144, Apr. 2019.
[62] Z. Che, S. Purushotham, K. Cho, D. Sontag, and Y. Liu, "Recurrent neural networks for multivariate time series with missing values," *Sci. Rep.*, vol. 8, no. 1, p. 6085, 2018.

Raymond Chiong is currently an Associate Professor with The University of Newcastle, Callaghan, NSW, Australia. His research focuses on using automated intelligent computing methods, including machine learning and optimization algorithms, to drive societal change and improve quality of life. He has published more than 200 papers in these areas and attracted over three million dollars in research and industry funding.

Dr. Chiong is also the Editor-in-Chief of the *Journal of Systems and Information Technology*, an Editor of *Engineering Applications of Artificial Intelligence*, and an Associate Editor of *Engineering Reports*.

Zongwen Fan received the B.S. and M.Sc. degrees from Huaqiao University, Quanzhou, China, in 2014 and 2017, respectively, and the Ph.D. degree from The University of Newcastle, Callaghan, NSW, Australia, in 2021.

Dr. Fan is currently a Visiting Academic Affiliate with The University of Newcastle, Callaghan, NSW, Australia. He is also a Lecturer with the College of Computer Science and Technology, Huaqiao University, Quanzhou. His main research interests include machine learning, data mining, and fuzzy computing.



Zhongyi Hu received the M.Sc. and Ph.D. degrees from the Department of Management Science and Information Systems, School of Management, Huazhong University of Science and Technology, Wuhan, China, in 2012 and 2015, respectively.

He is currently an Associate Professor with the School of Information Management, Wuhan University, Wuhan. His research focuses on machine learning techniques and predictive analytics. He has published over 50 peer-reviewed international journal articles in these areas.

Dr. Hu is also an Associate Editor of the *Journal of Systems and Information Technology*.

Sandeep Dhakal received the M.Sc. degree in artificial intelligence from the University of St Andrews, St Andrews, U.K., in 2011. He is currently pursuing the Ph.D. degree with The University of Newcastle, Callaghan, NSW, Australia.

His research interests include evolutionary game theory, agent-based modeling, and sentiment analysis.