Neural Computing and Applications (2022) 34:14387–14395
https://doi.org/10.1007/s00521-022-07234-0

ORIGINAL ARTICLE

Spatial-temporal attention-based convolutional network with text and numerical information for stock price prediction

Chin-Teng Lin[1] · Yu-Ka Wang[1] · Pei-Lun Huang[2] · Ye Shi[1] · Yu-Cheng Chang[1]

Received: 13 April 2021 / Accepted: 29 March 2022 / Published online: 5 May 2022

Abstract

In the financial market, the stock price prediction is a challenging task which is influenced by many factors. These factors include economic change, politics and global events that are usually recorded in text format, such as the daily news. Therefore, we assume that real-world text information can be used to forecast stock market activity. However, only a few works considered both text and numerical information to predict or analyse stock trends. These works used preprocessed text features as the model inputs; therefore, latent information in text may be lost because the relationships between the text and stock price are not considered. In this paper, we propose a fusion network, i.e. a spatial-temporal attention-based convolutional network (STACN) that can leverage the advantages of an attention mechanism, a convolutional neural network and long short-term memory to extract text and numerical information for stock price prediction. Benefiting from the utilisation of an attention mechanism, reliable text features that are highly relevant to stock value can be extracted, which improves the overall model performance. The experimental results on real-world stock data demonstrate that our STACN model and training scheme can handle both text and numerical data and achieve high accuracy on stock regression tasks. The STACN is compared with CNNs and LSTMs with different settings, e.g. a CNN with only stock data, a CNN with only news titles and LSTMs with only stock data. CNNs considering only stock data and news titles have mean squared errors of 28.3935 and 0.1814, respectively. The accuracy of LSTMs is 0.0763. The STACN can achieve an accuracy of 0.0304, outperforming CNNs and LSTMs in stock regression tasks.

Keywords Long short-term memory · Convolutional neural network · Attention network · Stock price prediction

1 Introduction

Stock market prediction aims to forecast the future value of a stock based on historical stock prices and other information. Accurate prediction of the stock price is a challenging task due to its nonlinear and chaotic nature. It is affected by many factors, such as economic change, politics, and other global circumstances [1–3]. Deep learning (DL) techniques, recently, have been applied to many different fields to surpass human-level performance, including computer vision [4, 5], speech recognition [6, 7], autonomous control [8, 9] and Stock market prediction [10–15]. A recent survey on the applications of the DL for stock market predictions [10] revealed that the majority of studies used features such as historical prices and technical indicator data as inputs; only 0.83% of these studies included historical prices, fundamental and text data as inputs. Some of the studies that applied the DL techniques

✉ Yu-Cheng Chang
Yu-Cheng.Chang@uts.edu.au

Chin-Teng Lin
Chin-Teng.Lin@uts.edu.au

Yu-Ka Wang
YuKai.Wang@uts.edu.au

Ye Shi
Ye.Shi-1@uts.edu.au

[1] CIBCI lab, Australian Artificial Intelligence Institute, University of Technology Sydney, Sydney, NSW 2007, Australia

[2] Institute of Electrical and Control Engineering, National Chiao Tung University, Hsinchu 300093, Taiwan

on numerical information are listed in [11–15]. Singh and Srivastava proposed a market prediction model using principal component analysis to extract features as the input of a deep neural network (DNN) [11]. Hiransha et al [12] used a convolutional neural network (CNN) for stock price prediction based on historical prices and concluded that the convolutional sliding window can capture the activity of stock, which increased the prediction performance of the model.

In addition to CNNs, recurrent neural networks (RNNs) are also considered effective models in stock price prediction. Yu and Yen [13] proposed a stock prediction model based on long short-term memory (LSTM) networks. Liu et al. [14] proposed attention-based RNNs for multivariate time series prediction and applied it to stock price prediction. Both CNNs and RNNs can predict stock activity with acceptable accuracy by capturing spatial and temporal information, respectively, from numerical data. Eapen et al. [15] combined multiple pipelines consisting of a CNN and a bidirectional LSTM architecture to extract spatial-temporal features of stock market data and improved the prediction accuracy.

Apart from using only numerical information as an input, some researchers also leveraged non-numerical data to achieve precise stock market predictions; many approaches proposed simultaneously considering historical stock prices and event-specific information. Li et al. [16] used both stock indices and the daily news to generate sequential snapshots as inputs of a deep learning model constructed by a two-layer LSTM and a fuzzy connected layer. This deep learning model improved stock prediction with data from the Hong Kong Stock Exchange. In addition to daily news, other numerical data, such as the public mood from social media, were used as an input source for stock prediction. Bollen et al. [17] showed that the Dow Jones Industrial Average (DJIA) was related to specific public mood dimensions from Twitter feeds. This study used public mood and DJIA closing values as features to feed into a self-organising fuzzy neural network (SOFNN) model to predict future DJIA values. Huang et al. [18] proposed an algorithm to fuse multi-sourced data collected from internet news, social media and historical stock activity; the generated features were further fed to an LSTM model to predict stock fluctuations. Park, Leung and Ma [19] developed an information fusion model to link Twitter sentiments to stock prices. The fused information was used to improve prediction in vector autoregression models. Another integrated approach was presented in [20], which utilised market news and stock prices to predict stock price by using multi-kernel support vector regression (MKSVR) models. LSTM was applied in stock market price trend prediction in [21], an inclusion LSTM model sought to obtain the sequential information of the stock market. Another recent study [22] demonstrated the advantages of LSTM by using BiLSTM to generate the stock price movement representation in dealing with immaturity, non-linear instability and unreliability of the stock market.

In [10], the authors claimed that although text data are hard to collect and process, sentiment analysis applied on text data is potentially beneficial. According to the above literature survey, non-numerical information, such as daily news, can boost the accuracy of stock activity prediction. However, the extracted text features were not guaranteed to be highly relevant to the stock values; complicated information, i.e. both stock prediction information and a large amount of unrelated information and noise, may potentially decrease the performance of the prediction since the relationships between text features and stock values were not considered [23]. Therefore, several text-processing methods have been developed to discover hidden relationships in non-numerical information. Zhang et al. [24] proposed a sentiment analyser and an event extraction that were used to process social media and internet news data, respectively. The captured events and sentiment data were further integrated with quantitative stock data to form a feature map for stock price prediction. Akita et al. [25] adopted the paragraph vector method proposed by Le and Mikolov [26] to extract text features. Then, the text patterns were combined with numerical information as the input for stock trend prediction by LSTM.

LSTM sequentially accumulates information over a sentence, and this recurrent architecture can capture the relationships among text in sentences [27–30]. Thus, this paper presents a sequence-to-sequence autoencoder for text processing, capturing the relationships between every two texts in a sentence and generating representative information for each sentence. We then apply a convolution operation to extract the spatial feature maps of daily news. Besides, we also consider temporal factors into extracted feature maps by introducing an attention mechanism [14, 31–33], which has been proven to be very helpful in adaptively extracting the most relevant features. This paper proposes a spatial-temporal attention-based convolutional network (STACN). This multi-modal network uses an attention mechanism, a CNN and LSTM to extract spatial-temporal information from daily news and historical news stock activity for stock price prediction. The proposed STACN network uses both numerical and text data, preventing the loss of numerical stock information and decreasing the negative effects of noise from news information.

2 Proposed approach

This section describes the proposed approach in detail. The whole STACN architecture is presented in Fig. 1. The STACN consists of a CNN fed with non-numerical information, LSTMs fed with numerical information and a spatial-temporal attention network (STAN). The CNN includes convolution layers that can extract spatial feature maps from the input data. We feed spatial feature maps and the hidden states of LSTMs into the STAN to produce spatial-temporal attention weights for each feature map extracted by the convolution layers.

The detailed description of the CNN architecture is presented in Table 1. This CNN framework consists of four convolution blocks and three fully connected layers. The kernel size of each convolution layer is 5×5; in other words, the dimensions will be [1, 5, 5, last dimension number of past input maps]. Strides for kernels are [1, 1, 4, 1]. A maximum pooling layer is followed by the convolution layer, whose pooling kernel size is [1, 3, 3, 1], and stride pooling is [1, 1, 2, 1]. The number of maps in the CNN output is 128. Padding for all convolution layers is set as "same" to ensure that the output dimension and input dimension are identical. The numbers of hidden layers for the CNN fully connected (FC) layer are 90, 5, and 1, which correspond to the first, second, and third hidden layers, respectively.

2.1 Preprocessing using sequence-to-sequence autoencoder

In this study, we propose a preprocessing workflow to convert non-numerical information, i.e. daily news titles, into a map form for the CNN to predict stock values. The entire preprocessing flow is shown in Fig. 2. We use the SPACY tool for word embedding to convert words to 300-dimensional vectors. Because words are considered a time step for natural language processing, a sequence-to-sequence model is trained to convert sentences to thought vectors. Afterwards, these thought vectors are organized into map forms.

The proposed sequence-to-sequence autoencoder is constructed by LSTM [34], as shown in Fig 3. LSTM, which can be considered a deformation of the recurrent neural network, is widely used for tasks involving time

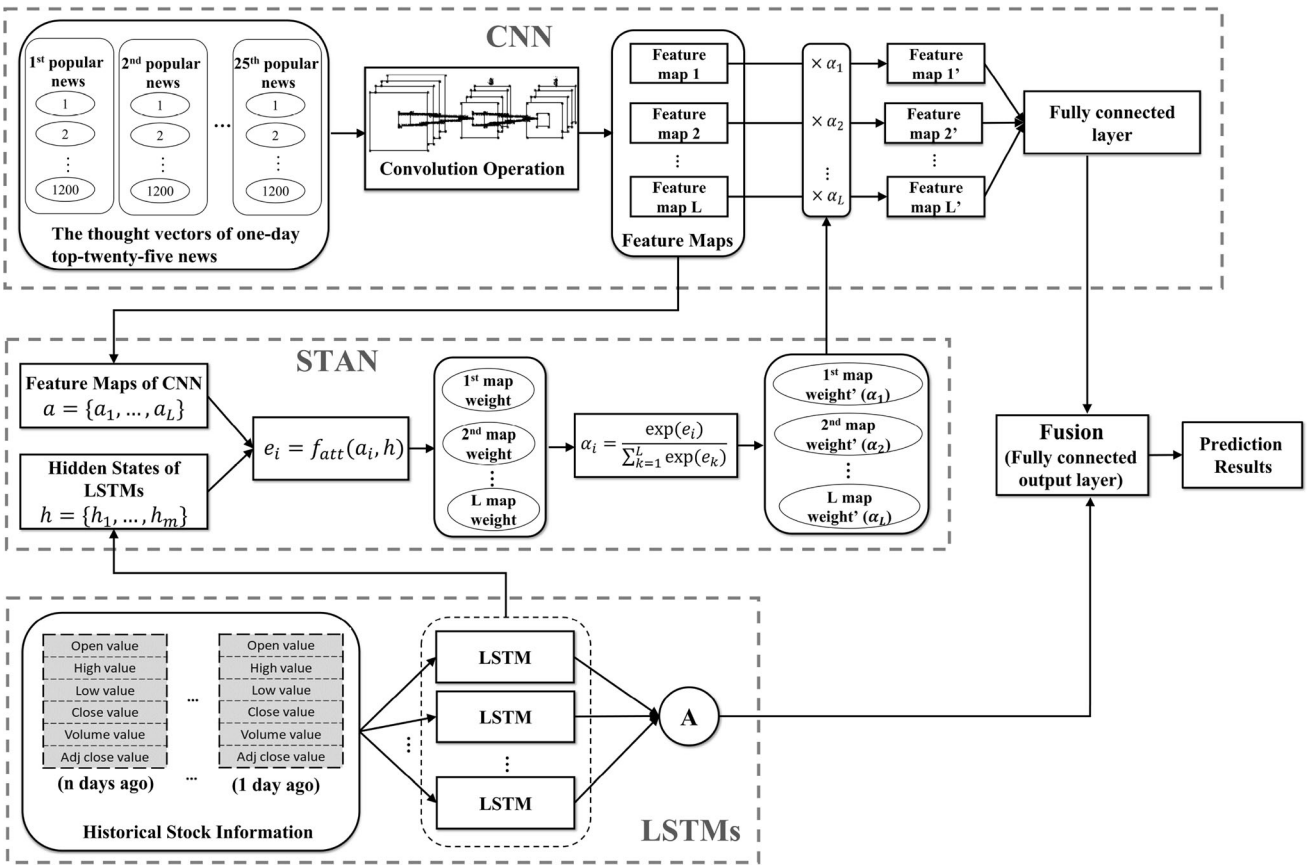

Fig. 1 The spatial-temporal attention-based convolutional network (STACN), which consists of a CNN, a STAN and a fusion network to generate the final output

Table 1 CNN architecture

	Filters/Neuron	Kernel size	Stride size	Padding	Activation
Conv3d 1	128	5×5	[1, 1, 4, 1]	Same	ReLU
Maxpooling 1		[1, 3, 3, 1]	[1, 1, 2, 1]	Same	
Conv3d 2	128	5×5	[1, 1, 4, 1]	Same	ReLU
Maxpooling 2		[1, 3, 3, 1]	[1, 1, 2, 1]	Same	
Conv3d 3	128	5×5	[1, 1, 4, 1]	Same	ReLU
Maxpooling 3		[1, 3, 3, 1]	[1, 1, 2, 1]	Same	
Conv3d 4	128	5×5	[1, 1, 4, 1]	Same	ReLU
Maxpooling 4		[1, 3, 3, 1]	[1, 1, 2, 1]	Same	
FC 1	90				ReLU
FC 2	5				ReLU
FC 3	1				ReLU

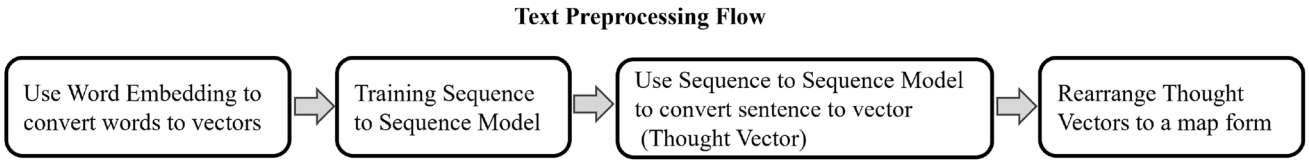

Fig. 2 Text preprocessing flow for feature map generation

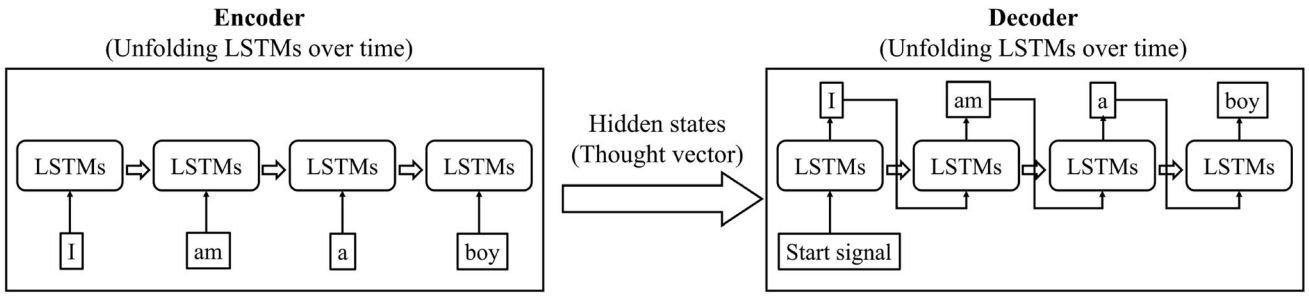

Fig. 3 Sequence-to-sequence autoencoder. We construct the encoder and decoder parts of the model by using LSTM. The two parts have identical numbers of hidden neurons and layers

series and words. Figure 4 shows the architecture of an LSTM unit. The hidden states h_t are memory cell output vectors controlled by an output gate o_t to weight the hyper-tangent of store value C_t. Forget gate f_t determines the percentage of past store values C_{t-1} to reserve. Input gate i_t determines the impact of cell input vector $\tilde{C}_t$ on store value

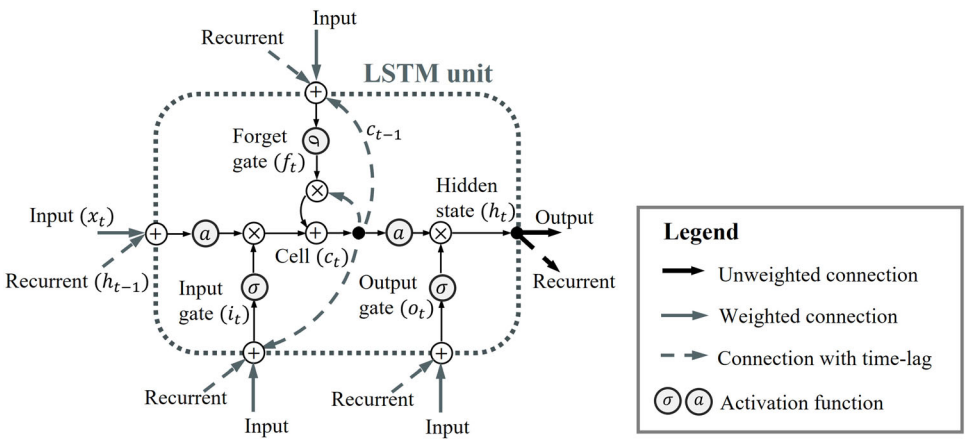

Fig. 4 An LSTM unit, which can be considered a deformation of a recurrent neural network

C_t. We set the initial values of cell state vector C_t and hidden states h_t to 0. Variable x_t is the input vector; h_t is a vector of hidden states (i.e. the memory cell output vector); and a and σ are bipolar and hyperbolic activation functions, respectively.

The sequence-to-sequence autoencoder includes an encoder, hidden neurons (hidden states), and a decoder, as shown in Fig. 3. The input layer of the encoder and output layer of the decoder are constructed by using LSTM units. The encoder and decoder have identical numbers of hidden neurons and layers. The encoder is sequentially fed words of a news title (one word is considered one timestamp), while the decoder sequentially generates the same words by taking the previous output as the input. The hidden states represent the thought vector, which is passed to the decoder part to serve as the initial hidden states instead of zero.

We use the adaptive moment estimation (ADAM) method for optimization [35], and its label is the input news title. When a training process is completed, the thought vectors represent a news title. We select the top 25 most important news feature maps. A kernel with the corresponding 2D weights will calculate the dot product of 2D input feature maps or input data along the vertical and horizontal directions and subsequently assign the sum of the products as a pixel value. In a CNN, there can be many convolutional layers. Between successive convolutional layers, pooling layers are used to reduce the spatial size of the feature map and enhance the characteristic features using the max operation. Finally, the outputs from the last feature maps are flattened to 1D vectors and connected to the fully connected layers. In our approach (Ref. Fig. 4), the 2D convolution kernel is extended to a 3D kernel. The 3D convolutional operation is implemented as follows:

$$v_{ij}^{xy} = \text{relu}\left(b_{ij} + \sum_{m}\sum_{p=0}^{P_i-1}\sum_{q=0}^{Q_i-1} w_{ijm}^{pq} v_{(i-1)m}^{(x+p)(y+q)}\right) \quad (1)$$

where $\text{relu}(\cdot) = \max(0, \cdot)$ is the rectified linear activation unit; v_{ij}^{xy} is the unit at the x and y position of the feature map between the i-th and j-th layers; b_{ij} is the bias between the two layers; m is the m-th filter in the convolutional layer; P and Q are the kernel sizes of the filters; and w_{ijm}^{pq} is the weight at the p and q position of the m-th filter.

2.2 Spatial-temporal attention network

Since thought vectors transfers from news titles include both stock-related features and a large amount of unrelated information and noise, importing all features from the news into the model can degrade the prediction. Therefore, we propose a STAN to identify the relevant content for stock prediction (Ref. Fig. 1). The STAN leverages LSTM to extract hidden temporal relationships in stock data. We feed a series of past stock values into LSTMs to obtain the temporal information of stock value changes. The hidden states of LSTMs $h = \{h_1, h_2, \cdots, h_m\}$ and the feature maps generated by the last convolutional layers of the CNN are fed into the attention neural network. The output of the attention neural network is the feature map weighting values $\alpha = \{\alpha_1, \alpha_2, \cdots, \alpha_L\}$. The weighting values can emphasize the importance of particular features via multiplication.

2.3 Spatial-temporal attention-based convolutional network

We combine the CNN, STAN and fusion network to construct a spatial-temporal attention-based convolution network (STACN); then, we use the fusion network to generate the final output, as shown in Fig. 1.

The workflow is as follows. First, we arrange the thought vectors of text as a map to serve as the input for the CNN. Next, we sequentially feed the concatenated stock numbers into the LSTMs to derive the hidden states to gather information on stock trends. The LSTM output is produced by an ReLU neuron that is presented as node A in Fig. 1. The CNN will generate feature maps layer by layer via the convolution operation. We utilize the last feature maps and hidden states of the LSTMs to produce the weighting value for each feature map. Then, we multiply each feature map by its corresponding weighting value and flatten the maps into the fully connected layer of the CNN. Finally, we use the fusion layer to combine the LSTM outputs with the CNN to generate predictions. We adopt the ADAM optimization to optimize the hyperparameters of the STAN and CNN. The formulas of the STACN are shown below.

$$P_{\text{CNN}} = f_{\text{CNN}}(a \odot \alpha) \quad (2)$$

$$P_{\text{LSTM}} = f_{\text{LSTM}}(h) \quad (3)$$

$$P_{\text{Result}} = f_{\text{Fusion}}(P_{\text{CNN}}, P_{\text{LSTM}}). \quad (4)$$

Here, f_{CNN}, f_{LSTM} and f_{Fusion} are multilayer perceptrons, P_{CNN}, P_{LSTM} and P_{Result} are their corresponding model outputs, $a = a_1, a_2, \cdots, a_L$ denotes the features from a lower convolutional layer of the CNN, $h = \{h_1, h_2, \cdots, h_m\}$ is the hidden state of the LSTM, $\alpha = \{\alpha_1, \alpha_2, \cdots, \alpha_L\}$ is the weighting value for the feature maps

in the last layer of the CNN, and $\odot$ represents the dot multiplication. In the STACN, $L \in \mathbb{R}$, $m = 5$ and $L = 128$.

3 Experiments

3.1 Data description

The dataset used in this experiment includes the DJIA historical data and news titles. The DJIA historical data were collected from Yahoo Finance from 2008/08/08 to 2016/07/01, and the news titles were from the Reddit WorldNews channel from 2008/06/08 to 2016/07/01. To keep the considered time ranges of stock and news data identical, the news data are truncated to 2008/08/08 to 2016/07/01. The DJIA contains six indicators for each day: open value, high value, low value, close value, volume, and adj. close value (adjusted closing price). The news data for a single date include 25 headlines, which are selected and ranked in terms of popularity according to the votes of Reddit users. The total number of samples is 1980, including 1200 training samples and 780 test samples. We divide the test samples into 26 batches of 30 samples. The first test batch lacks ten-day-ahead historical stock data from 2008/06/08. We therefore record the prediction performance with and without test batch 1 for a fair comparison. For our implementation, we use the news on the predicted day as our model input and perform a regression to determine the close value/1000.

3.2 Model settings

Text preprocessing For the sequence-to-sequence autoencoder, each news title is truncated or zero-padded to 71 representative words. The input and output dimensions are set to 300. There are four layers, and each layer has 300 LSTMs for both the encoder and decoder. We select 1700 training samples from the news data and adopt ADAM optimization. The learning rate is 0.001 [36, 37], and 1000 training epochs are used, which are enough for the sequence-to-sequence autoencoder to converge at a stable training performance. The activation function is ReLU. The loss function is the mean squared error (MSE) between the input and output of the sequence-to-sequence autoencoder.

The proposed stock activity prediction model For the STACN, the inputs for the CNN are 3D maps. The first dimension is set to 25 for the number of news titles each day. The second dimension is set to 300 for the thought vector columns, which is identical to the number of LSTMs in each layer of the encoder or decoder of the sequence-to-sequence autoencoder. The third dimension is set to 4 to represent a row of thought vectors that correspond to the number of hidden layers of the encoder or decoder of the sequence-to-sequence autoencoder. Therefore, considering the batch size, the input dimensions will be [batch size, 25, 300, 4]. Thus, the length of a whole extracted though vector is $300 \times 4 = 1200$. The CNN kernel is a 5×5 window; in other words, the dimensions will be [1, 5, 5, last dimension number of past input maps]. The strides for the kernel are [1, 1, 4, 1]. Maximum pooling with a kernel size of [1, 3, 3, 1] and stride pooling of [1, 1, 2, 1] are adopted. The number of convolutions (with pooling) is one. The number of feature maps in the CNN output is 128. The numbers of hidden layers in the CNN fully connected layer are 90, 5, and 1, which correspond to the first, second, and third hidden layers, respectively. For LSTMs in the STACN, the number of hidden neurons in the first hidden layer is 5, and the second hidden layer has 1 original neural network node. There are 10-time steps. The attention learning part of the STAN has 2 hidden layers with 100 and 50 nodes for the first and second layers, respectively. To fuse the outputs of the LSTMs and the CNN, a fully connected layer is built using one hidden layer with three nodes. The loss function of this layer is the MSE. The time steps are set to 10 to consider the numerical information for the last 10 days. The inputs of the STACN were obtained via normalization. The environment is TensorFlow operated by Python 3 on Linux.

Different architectures for stock activity prediction We conduct experiments with various architectures of stock activity prediction models for comparative purposes and investigate the ability of the proposed STACN model. As shown in Table 2, we employed CNNs and LSTMs to perform stock price regression task; two CNNs were fed with either historical DJIA stock data or 25 headlines of daily news, respectively, and the LSTMs were fed with historical DJIA stock. To train the model considering only one type of input, we use T-S-1, which denotes training strategy 1 to training the CNNs and LSTMs. The optimiser is ADAM with a 0.001 learning rate, and the number of training epochs is set to 4000. The proposed STACN model which receives both historical DJIA stock value or 25 headlines of daily news was trained by using T-S-2, which denotes training strategy 2. T-S-2 includes two training phases; phase 1 is a pre-training process of the LSTMs, and phase 2 is the training of the whole STACN. The ADAM optimiser is used, and the number of training epochs is set to 2000 in both phase 1 and 2. We also compare the prediction performance between the STACN with and without attention to verify the effect of the STAN. The two STACNs used the identical configuration in training process. The DJIA stock data are rearranged into maps, whose rows represent n days ago, e.g. $n = 10$, and whose columns

Table 2 Training strategies for different structures of stock price regression models

Type	Description	Training epochs	Models
T-S-1	Training whole model	4000 epochs	CNN (Only Stock),CNN (Only News),LSTMs (Only Stock)
T-S-2	Phase 1: Pretraining LSTMs	Phase 1: 2000 epochs	STACN, STACN without Attention
	Phase 2: Training whole model	Phase 2: 2000 epochs	

Table 3 Comparison of different structures in stock price regression task. MSE and r represent the mean squared error and the correlation coefficient, respectively

	CNN (only news)	CNN (only stocks)	LSTMs (only stocks)	STACN without Attention	STACN
Training Strategy	T-S-1	T-S-1	T-S-1	T-S-2	T-S-2
MSE (Total)	29.7366	0.1953	0.1255	0.111	**0.0486**
MSE (No batch 1)	28.3935	0.1814	0.0763	0.082	**0.0304**
r (Total)	0.3375	0.9552	0.9560	0.9358	**0.9765**
r (No batch 1)	0.3159	0.9710	0.9642	0.9422	**0.9847**

The bold is to emphasise the performance (MSE) made by the proposed STACN

are the six indicators previously described: open value, high value, low value, close value, volume, and adj. close value (adjusted closing price). Therefore, the stock data serve as input in the shape of [batch size, n days = 10, indicators = 6, 1]. In addition, we concatenate 25 thought vectors of daily news headlines as a one-day feature, of which the dimension is [batch size, 25, 300, 4].

3.3 Prediction performance comparison

We compare the MSE and the correlation coefficient (r) of the STACN and other structures in the stock price regression task. According to Table 3, the CNN with only news headlines has the worst performance (28.3935 MSE without batch 1); this low performance is due to the weak time-relevance among daily news titles. In contrast, the CNN with historical stock data results in better prediction performance (0.1814 MSE without batch 1) because past stock values are more relevant to stock regression than news titles. In addition, the LSTMs with historical stock data outperform the CNNs (0.0763 MSE without batch 1), which implies that the such recurrent structure is helpful to catch time-relevance information in the stock activity. The proposed STACN model that considers numerical information and news titles outperforms other structures; this models MSE is 0.0486 and 0.0304 with and without test batch 1, respectively. The STACN leverages time-relevant states of LSTMs and spatial relationships obtained by the convolution operation to weight useful information in the extracted feature maps of the CNN. This mechanism effectively increases the performance in the stock regression task.

3.4 Attention method effect

To investigate the effect of attention learning in the STACN, we remove the attention learning part from the STACN, i.e. only the CNN, LSTMs and fusion network are used to perform stock prediction. We call this method "STACN without attentio". We compare the STACN with the STACN without attention using the MSE. As shown in Table 3, the STACN is better than the STACN without attention in the regression task; removing the attention network from the STACN will degrade the performance of prediction.The prediction performance is similar to the LSTMs because the convolution layers in the STACN cannot obtain stock-related information from the news titles; the LSTMs in the STACN contribute the most of positive effects in prediction accuracy. In contrast, the STACN exploits spatial-temporal attention generated by the STAN that can enhance the effect of stock-related information and filter out those irrelevant information so that the extracted feature maps can be effectively used to increase the prediction performance.

4 Conclusion

This study proposes a novel deep learning model and procedure to process both numerical and text information for stock prediction. The proposed STACN combines the advantages of a CNN, LSTM, and an attention learning model. The CNN can evaluate the semantic composition of text and capture the spatial relationships among the top-twenty-five piece of daily news in one day. The LSTM can

identify the previous hidden trends of stock information; it provides the attention learning model with temporal information. The attention learning model can strengthen the significant feature maps for the CNN. Furthermore, a sequence-to-sequence autoencoder is developed to convert the news titles to thought vectors, which are rearranged to a map form to feed into the CNN. We compare the prediction performance of three different architectures of the STACN to verify the effect in considering news titles, stock values and both, respectively. The CNN, which is fed with only news titles, has the worst prediction performance because it cannot strengthen the stock-related feature. The CNN which considers only past stock values has a better prediction performance than the CNN considering new titles only because past stock values are more relevant to stock regression than news titles. The LSTMs fed with historical stock data outperform the CNNs because it can capture time-relevance information in the stock activity. The proposed STACN which receives both news titles and past stock values as inputs has the best performance in stock prediction. The experiment results demonstrate that using both news and numerical information can improve the accuracy of stock prediction. We further compared against the STACN without attention, which demonstrated that the spatial-temporal attention generated by the STAN that can enhance the effect of stock-related information and filter out those irrelevant information. We are eager to extend the STACN to various fields including computer vision, speech processing, and physiological signal processing. Multiple types of input (i.e. temporal data, spatial data, or multidimensional data) can be considered. We look forward to improving the performance of the method and building more general applications in future research.

Acknowledgements This work was supported in part by the Australian Research Council (ARC) under discovery grants DP180100670 and DP180100656. The research was also sponsored in part by the Australia Defence Innovation Hub under Contract No. P18-650825 and the US Office of Naval Research Global under Cooperative Agreement Number ONRG-NICOP-N62909-19-1-2058. We also thank the NSW Defence Innovation Network and NSW State Government of Australia for financial support of part of this research through grant DINPP2019 S1-03/09.

Funding Open Access funding enabled and organized by CAUL and its Member Institutions.

Declarations

Conflict of interest The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Thakkar A, Chaudhari K (2021) Fusion in stock market prediction: a decade survey on the necessity, recent developments, and potential future directions. Information Fusion 65:95–107
2. Zheludev I, Smith R, Aste T (2014) When can social media lead financial markets? Scientific Rep 4(1):1–12
3. Moat HS, Curme C, Avakian A, Kenett DY, Stanley HE, Preis T (2013) Quantifying wikipedia usage patterns before stock market moves. Scientific Rep 3(1):1–5
4. Masone C, Caputo B (2021) A survey on deep visual place recognition. IEEE Access 9:19516–19547
5. Liu W, Wang Z, Liu X, Zeng N, Liu Y, Alsaadi FE (2017) A survey of deep neural network architectures and their applications. Neurocomputing 234:11–26
6. Padmanabhan J, Johnson Premkumar MJ (2015) Machine learning in automatic speech recognition: a survey. IETE Technical Rev 32(4):240–251
7. Malik M, Malik MK, Mehmood K, Makhdoom I (2021) Automatic speech recognition: a survey. Multimedia Tools Appl 80(6):9411–9457
8. Sun Z, Zou J, He D, Zhu W (2022) Path-tracking control for autonomous vehicles using double-hidden-layer output feedback neural network fast nonsingular terminal sliding mode. Neural Comput Appl 34(7):5135–5150
9. Kuutti S, Bowden R, Jin Y, Barber P, Fallah S (2020) A survey of deep learning applications to autonomous vehicle control. IEEE Transactions Intell Transport Syst 22(2):712–733
10. Jiang W (2021) Applications of deep learning in stock market prediction: recent progress. Expert Syst Appl 184:115537
11. Singh R, Srivastava S (2017) Stock prediction using deep learning. Multimedia Tools Appl 76(18):18569–18584
12. Hiransha M, Gopalakrishnan EA, Menon VK, Soman K (2018) Nse stock market prediction using deep-learning models. Procedia Computer Sci 132:1351–1362
13. Yu P, Yan X (2020) Stock price prediction based on deep neural networks. Neural Comput Appl 32(6):1609–1628
14. Liu Y, Gong C, Yang L, Chen Y (2020) Dstp-rnn: a dual-stage two-phase attention-based recurrent neural network for long-term and multivariate time series prediction. Expert Syst Appl 143:113082
15. Eapen J, Bein D, Verma A (2019) Novel deep learning model with cnn and bi-directional lstm for improved stock market index prediction In: 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC), pp 0264–0270 IEEE
16. Li X, Wu P, Wang W (2020) Incorporating stock prices and news sentiments for stock market prediction: a case of hong kong. Information Process Manag 57(5):102212
17. Bollen J, Mao H, Zeng X (2011) Twitter mood predicts the stock market. J Comput Sci 2(1):1–8
18. Huang J, Zhang Y, Zhang J, Zhang X (2018) A tensor-based submode coordinate algorithm for stock prediction In: 2018 IEEE

Third International Conference on Data Science in Cyberspace (DSC), pp 716–721 IEEE
19. Park J, Leung H, Ma K (2017) Information fusion of stock prices and sentiment in social media using granger causality In: 2017 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI), pp 614–619. IEEE
20. Li X, Huang X, Deng X, Zhu S (2014) Enhancing quantitative intra-day stock return prediction by integrating both market news and stock prices information. Neurocomputing 142:228–238
21. Shen J, Shafiq MO (2020) Short-term stock market price trend prediction using a comprehensive deep learning system. J Big Data 7(1):1–33
22. Long J, Chen Z, He W, Wu T, Ren J (2020) An integrated framework of deep learning and knowledge graph for prediction of stock price trend: an application in chinese stock exchange market. Appl Soft Comput 91:106205
23. Li X, Xie H, Chen L, Wang J, Deng X (2014) News impact on stock price return via sentiment analysis. Knowl Based Syst 69:14–23
24. Zhang X, Qu S, Huang J, Fang B, Yu P (2018) Stock market prediction via multi-source multiple instance learning. IEEE Access 6:50720–50728
25. Akita R, Yoshihara A, Matsubara T, Uehara K (2016) Deep learning for stock prediction using numerical and textual information In: 2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS), pp 1–6 IEEE
26. Le Q, Mikolov T (2014) Distributed representations of sentences and documents In: International Conference on Machine Learning, pp 1188–1196
27. Chen T-L, Cheng C-H, Teoh H-J (2008) High-order fuzzy time-series based on multi-period adaptation model for forecasting stock markets. Physica Statistical Mech Appl 387(4):876–888
28. Dixon M (2018) Sequence classification of the limit order book using recurrent neural networks. J Comput Sci 24:277–286
29. Zazo R, Lozano-Diez A, Gonzalez-Dominguez JT, Toledano D, Gonzalez-Rodriguez J (2016) Language identification in short utterances using long short-term memory (lstm) recurrent neural networks. PloS One 11(1):0146917
30. Li H, Shen Y, Zhu Y (2018) Stock price prediction using attention-based multi-input lstm In: Asian Conference on Machine Learning, pp 454–469 PMLR
31. Xu K, Ba J, Kiros R, Cho K, Courville A, Salakhudinov R, Zemel R, Bengio Y (2015) Show, attend and tell: neural image caption generation with visual attention In: International Conference on Machine Learning, pp 2048–2057
32. Wang Z, Hamza W, Florian R (2017) Bilateral multi-perspective matching for natural language sentences arXiv preprint arXiv:1702.03814
33. Chopra S, Jain S, Sholar JM (2017) Towards automatic identification of fake news: headline-article stance detection with lstm attention models In: Stanford CS224d Deep Learning for NLP Final Project
34. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780
35. Kingma DP, Ba J (2014) Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980
36. Yi D, Ahn J, Ji S (2020) An effective optimization method for machine learning based on adam. Appl Sci 10(3):1073
37. Kim T, Kim HY (2019) Forecasting stock prices with a feature fusion lstm-cnn model using different representations of the same data. PloS One 14(2):0212320