future internet

MDPI

Article

Rumor Detection Based on Attention CNN and Time Series of Context Information

Yun Peng * and Jianmei Wang

School of Computer Information and Engineering, Jiangxi Normal University, Nanchang 330022, China; jianmei@jxnu.edu.cn

* Correspondence: pengyun@jxnu.edu.cn

Abstract: This study aims to explore the time series context and sentiment polarity features of rumors' life cycles, and how to use them to optimize the CNN model parameters and improve the classification effect. The proposed model is a convolutional neural network embedded with an attention mechanism of sentiment polarity and time series information. Firstly, the whole life cycle of rumors is divided into 20 groups by the time series algorithm and each group of texts is trained by Doc2Vec to obtain the text vector. Secondly, the SVM algorithm is used to obtain the sentiment polarity features of each group. Lastly, the CNN model with the spatial attention mechanism is used to obtain the rumors' classification. The experiment results show that the proposed model introduced with features of time series and sentiment polarity is very effective for rumor detection, and can greatly reduce the number of iterations for model training as well. The accuracy, precision, recall and F1 of the attention CNN are better than the latest benchmark model.

Keywords: rumor event detection; sentiment polarity; time series algorithm; attention CNN

Citation: Peng, Y.; Wang, J. Rumor Detection Based on Attention CNN and Time Series of Context Information. *Future Internet* **2021**, *13*, 267. https://doi.org/10.3390/fi13110267

Academic Editor: Paolo Bellavista

Received: 31 August 2021
Accepted: 18 October 2021
Published: 25 October 2021

1. Introduction

By the end of 2020, China had 989 million netizens and 70.4% Internet penetration, with more than 1.6 billion mobile Internet users. With the popularity of the Internet, microblog, WeChat and other applications gradually occupy people's lives. They become important platforms for publishing and collecting information. Taking Weibo as an example, on the one hand, Sina Weibo provides sentiment outlets for the public. On the other hand, it also forms a hotbed for making and disseminating rumors [1]. Rumors on social media are rampant and it is difficult to distinguish between credible and untrustworthy information, which can lead to social unrest and seriously endanger national security. Therefore, it is particularly important to detect rumors in the early development of rumors.

Rumor recognition has always concerned scholars. The existing rumor recognition models based on machine learning mainly deal with the content features, link features, character features and keyword features of a large number of microblog rumor texts manually. Then, these models comprehensively utilize user features and interactive features. The SVM model [2–4], Bayesian classifier [5] and improved Naive Bayes method [6] are used to identify rumors. In addition, deep text sentiment features [7] have been proven to be more effective in identifying rumors than other features. Z. Wang et al. [8] used the construction of sentiment dictionary to study the influence of the delicate emotion of events on rumor recognition. However, for the construction of sentiment dictionary, the manual work is heavy, and the accuracy of the sentiment dictionary also affects the identification of rumors. In fact, in the whole life cycle of rumors, the sentiment polarity of events is also changing, which is different from non-rumors.

Moreover, in order to mine the key features of dynamic rumors and enhance robustness, deep neural networks are favored by scholars. J. Ma et al. [9] used classical RNN to capture the continuity of microblog posts over time. Additionally, the sequence length was equal to the number of posts, which was expensive, ineffective and it did not learn the

deep features of the text. J. Wang et al. [10] used a time series algorithm to divide the whole rumor life cycle into 20 groups. At the same time, a CNN was used to learn the internal features between different life cycles and within different life cycles. At this time, the length of the input sequence is 20, which greatly reduces the amount of calculation. It also can reduce the resource occupancy rate and improve the accuracy of rumor recognition. However, the sentiment tendencies of posts are different in different life cycles of rumors. J. Wang et al. did not consider this feature and did not highlight the weight of each feature in different time series.

In order to solve the above two problems, we first use the time series (TS) algorithm to divide the time series data of events and then use the classical SVM model to obtain the sentiment polarity in different periods. Finally, the sentiment polarity is combined with the text vector and the fused event features are used as the input of the convolutional neural network for rumor recognition. At the same time, the spatial attention mechanism is added to the model to adjust the weight, so as to focus on more important input features and enhance the effect of rumor recognition. The contributions of this study are as follows:

(1) We verify that sentiment polarity features in different time series can effectively identify rumors.
(2) We verify that adding an attention mechanism can effectively adjust the weight of rumor features in different time series, focus on more important features and reduce the number of iterations for model training greatly.
(3) We realized the convolutional neural network model with the attention mechanism added by combining sentiment polarity features and time series information. Additionally, we verified that the improved model can greatly reduce the number of iterations of model training through experimentation.

The rest of this paper is organized as follows: Section 2 discusses related work, followed by the problem statement and proposed solution designed in Section 3. The experiment and analysis are discussed in Section 4. Section 5 concludes the paper.

2. Related Works

The rumor event detection problem is generally regarded as a classification problem. A group of rumor events are marked as rumors (R) and conventional events are marked as non-rumors (NR). The machine learning algorithm is trained. The idea is to make an algorithm that can learn the features of rumor events from training. When a new event appears, it can recognize whether this event is a rumor. At present, the automatic detection methods of rumor events can be simply divided into two categories: methods based on traditional machine learning and methods based on deep learning.

2.1. Method Based on Traditional Machine Learning

The traditional machine learning methods for rumor recognition are mainly reflected in feature extraction and algorithm improvement. T. Takahashi et al. [11] analyzed the explosiveness of Twitter events, post forwarding rate and vocabulary distribution differences to calculate the credibility score. K. K. Kumar et al. [12] used user features, content features and event features to calculate user credibility, content credibility and event credibility, respectively. Additionally, then they used these features to calculate the Gini coefficient to identify rumors. Recently, scholars still use the minimum credibility to study rumor clarification. The scholar proposed the Longest-Effective-Hops (LEH) algorithm to solve the problem of finding users with the minimum credibility in the shortest time [13]. The extraction of these features can effectively identify rumors, but the features of post forwarding rate and event features depend on a large number of manual collections. Considering the post forwarding situation is dynamic, it is difficult to extract these context features. Manually extracted rumor features cannot adapt to the complex environment of social development and change. H. Bingol et al. [14] comprehensively evaluated the recognition effect of basic models such as OneR (One Rule), Naive Bayes, ZeroR, JRip, Random Forest, Sequential Minimal Optimization and Hoeffding Tree. It is not enough to mine the shallow

features related to rumor posts, so E. Mao et al. [7], S. Luo et al. [15], G. Cai et al. [16] and Z. Zeng et al. [17] extracted the sentiment features of rumor texts and used random forest model to achieve good rumor recognition results. These verified that the sentiment features of rumor text can effectively identify rumors. However, these scholars only considered the static text features and sentiment features of rumors but did not consider that these features changed with the development of rumors.

Therefore, many scholars had paid attention to the time series features of rumors. J. Ma et al. [18] proposed SVM-TS model to identify rumors. This method used time series modeling technology to interpret varied social background information (post content information, user information, dissemination information), so as to capture the time-varying characteristics of these features. Finally, an SVM model was used to identify rumors. Z.-H. Wang et al. [19] constructed the $SVM_{com}{}^{DTS}$ model to capture the dynamic sequence features of rumors changing with time. On this basis, three new features based on the communication theory of rumor events in sociology (post popularity, post ambiguity and post spread) are added to the $SVM_{all}{}^{DTS}$ model. However, these two scholars only considered the dynamic sequence features of rumor texts changing with time. They also did not consider the deep features of texts and the sentiment features of events in different time series.

All in all, the traditional machine learning method has a good effect on rumor recognition. However, on the one hand, although the machine learning method considers the rumor text features, user features, sentiment features and time series, they do not consider the deep features of the text changes with time and, furthermore, it is not adapted to the rumors of the times. On the other hand, the feature engineering of machine learning methods consumes a lot of manpower and material resources. The user features, content features, event features and time series features used in the above traditional machine learning method all need to be constructed manually. For example, these features are manually extracted including the number of users followed, the number of users following, user registration time, special symbols like "?" "!" and post text content. Then, they are used by classic or improved traditional machine learning models to train and recognize rumor. There is no doubt that such structural features are labor intensive. However, most models based on deep learning do not need to manually construct features. It directly collects posts related to the event. Then, the model based on deep learning vectorizes the obtained post information and automatically learns the high-level features hidden in it. The special symbols like "?" "!" do not need to be extracted manually. Therefore, deep learning methods are more and more favored by scholars.

2.2. Method Based on Deep Learning

In recent years, deep neural networks have become more and more popular and they have achieved good performance in many NLP tasks. Y. Xu et al. [20] used multiple RNNs to deeply mine dynamic time features and modeled the social background information of events as N equal time series, so as to obtain text sequence coding and capture the background information of relevant posts over time. On this basis, N. Ruchansky et al. [21] divided the whole rumor recognition model into three modules: Capture, Score and Integrate, in which the Capture module uses RNN to learn the time representation of the text. From the experimental results, their experimental results are better than previous work (SVM-TS [18], $SVM_{com}{}^{DTS}$ [19]) which prove the effectiveness of deep neural network model in rumor detection. The RNN was used to learn text content of rumors shows that the length of time series is equal to the number of posts and the calculation is too large. Later scholars used GRU [8] and its variant gated neural network model (PGNN) [22] to track the time and broadcast features of posts. Additionally, then they added attention mechanism to adjust the weight of each node which achieved better performance. However, the models adopted by these scholars only considered the dynamic time features of post propagation. They did not mine the deep text features inside the posts. P. Yin et al. [23] proposed a convolution-long-short memory network (C-LSTM) model based on the historical off-line

features of microblog users. The model combines the time feature and the internal feature of the post, but this is more complex. The number of input parameters is large and this model occupied a large number of resources. Some scholars have also improved the LSTM model [24,25] by combining word attention and context information, or mining the homogeneity and dialogue structure features of rumors which has achieved good results. However, the relationship between the different life cycles and the life cycle of the development of microblog events has not been considered. Additionally, the model is complex.

In addition, the propagation structure of rumors has also been proven to be effective in identifying rumors. T. Bian et al. [26] proposed a bi-directional graph convolution network (Bi-GCN) model to explore the two structural features of rumor propagation and diffusion by manipulating the top-down and bottom-up rumor propagation. Although the accuracy of the model reached 88.6% on the experimental data set, it has several GCN layers and the size of the input data set is the number of posts. The calculation of the whole model is relatively expensive. J. Ma et al. [27] and S. Kumar et al. [28] proposed a tree-like neural network model to learn the propagation features of rumors by tracking the non-sequential propagation structure of tweets or representing social media conversations as binary constituency trees. In rumor recognition, the F1-macro value of this model is higher than that of the best model at present which proves that the features of communication structure can effectively identify rumors. However, in this changing society, the propagation structure of rumors will also change. When the model that has learned the features of propagation structure faces new rumors, the effect may be greatly reduced.

The convolutional neural network model has produced good results in rumor recognition. Z. Liu et al. [29] and S. Santhoshkumar et al. [30] used the convolutional neural network (CNN) model to train and mine the internal text features of rumors. However, they did not take into account the time series features in the life cycle of rumors. The graph convolution neural network [31,32] derived from the convolution neural network and the improved model EGCN [33] convert the microblog rumor data to graph data. Then, they use the convolution neural network to train the labeled data. By updating the node weight in the graph, the information is passed to the unlabeled data which greatly reduces the workload of rumor data annotation. The variants of these CNN models have achieved good results in rumor recognition. However, the number of parameters of these models is the product of the output dimensions of the two hidden layers, Therefore, the calculation is expensive.

Z. Liu et al. [29] proposed the basic CNN model and used Doc2Vec [34] to vectorize microblog posts and set the dimension size of the convolution kernel as the length of the post vector. These operations can effectively extract features between posts, and they also can extract features within the posts. The model greatly reduces the labor intensity of artificial feature construction and learns the deep features between microblog posts. However, they do not consider the time series features of posts between different life cycles of rumors. Based on this, J. Wang et al. [10] improved the model by using the time series algorithm (DTS) to divide the input of the model into 20 groups according to the time series. The model considers the time series features of rumor development and the parameters are reduced from 59,138 $\times$ 50 to 20 $\times$ 50 which greatly reduces the amount of calculation. However, the sentiment polarity of the event is not considered, and the key input features are not highlighted, so the accuracy of model needs to be improved.

Compared with delicate sentiment features, sentiment polarity features are more stable in complex environments. M. Bharti et al. [35] and K. ZU et al. [36] extracted 13 features such as positive and negative words of tweets, the length of tweets and sentimental features. The CNN and TF-IDF models were used to classify rumors. The precision rate, recall rate and F1 value of these two experiments increased by about 4%, which proved that the sentiment polarity feature could effectively identify rumors. The novel two-layer GRU model proposed by Z. Wang et al. [37] extracted the fineness features of rumor events and verified it on the OSNs data set. The results showed that it was superior to the latest rumor

detection model [38]. However, this scholar used a new sentiment dictionary to capture the features of delicacy which has a large amount of manpower. With the changing of the social network environment, the sentiment features of delicacy are sometimes inferior to the sentiment polarity features, because, in the face of different types of events, the emotions reflected in the comments of netizens are different. For example, most Internet rumors about international security are satire and criticism; when the events are about the death of a well-known person, they are sad; when the events are about natural disasters in society, they are sympathetic. Additionally, the models with counting sentiment features are highly complex and they require large amounts of calculation.

Enhanced feature representation plays a positive role in rumor recognition. On the one hand, multi-layer GCN [39] can enhance short-term information and data representation, then it can effectively deal with the early content of rumors [40]. On the other hand, the method is by adding an attention mechanism to the model. H. Adel et al. [41] and Y. Yuan et al. [42] add an attention mechanism to convolutional neural networks to emphasize the weight of uncertain detection factor features and measure feedback mechanisms. H. Zhang et al. [43] and Q. Li et al. [44] proposed a multi-task learning framework to detect rumors. The model was divided into a shared layer and a task-specific layer. An attention mechanism was introduced in the rumor detection process to mine the subtle semantic features hidden in it. The former was used to add image features published with rumor posts and the latter was used to add user credibility features. The experimental results of these models show that adding attention mechanism can adjust the weight of input features and enhance feature representation so that these models can effectively identify rumors. However, the enhanced features of the above model do not consider the sentiment polarity of the event. Additionally, they do not consider the time series features of rumor events and the addition of multiple task layers hides the increase in model complexity and the number of parameters.

On the basis of the above analysis, the traditional machine learning method and deep learning method cannot take into account the input parameters of the model. The input size of traditional machine learning and deep learning methods is mostly $n \times k$, where n is the number of posts for rumors or non-rumor events and k is the feature length encoded by each post. There is no doubt that this input is huge and variable, as the number of posts for events is roughly dozens to the tens of thousands. Therefore, the input parameters of these models are huge. These models also do not consider the time series features of rumors and sentiment polarity. We use the time series algorithm to divide the rumor text into several time series. The length of the model input is equal to the number of time series which greatly reduces the number of parameters of the model and extracts the sentiment polarity features in different time series. Then, the attention mechanism is added to the convolutional neural network to adjust the weight of the input features and enhance the relevant feature representation of the text, so as to effectively identify rumors.

3. Problem Statement and Proposed Solution

In order to consider the sentiment features of the text hidden in the time series grouping and weight the text features in different life cycles of rumors, this paper proposes CNN-TS + Sentiment + Attention (CTSA) model.

3.1. Problem Statement and Model Framework

Nowadays, once rumors appear on social media, posts related to rumor events can be easily searched according to keywords. The content of these posts describes the central content of rumor events better. However, the forwarding and comments of relevant posts cause the rapid spread of rumors. The research object of this paper is rumor events. The purpose of the experiment is to detect whether the event on a microblog is a rumor, rather than whether the post related to the event is a rumor. Once an event on the microblog is detected as a rumor, the post related to the event is also regarded as a rumor. For example, "# Qiqihar flood # Heilongjiang Qiqihar has also had a serious flood. Watching the videos

taken by netizens is really serious. I hope you will pay more attention and support!" This rumor and the related microblog posts "are there any rescue ?" and "look pathetic / / @ Qi Yue July: are there any rescue ?" constitute the Qiqihar flood event. For this experiment, we just detect whether the event is a rumor and do not care whether the related posts are rumors. Therefore, the relevant symbols and explanations are given for the convolution neural network model integrating sentiment features and the attention mechanism.

Definition 1. *Rumor events [9]. Define all microblog event sets* $E = \{E_i\}$, *where* $E_i = \{m_{i,j}\}$, E_i *refers to the ith event, including all microblog posts related to it and* $m_{i,j}$ *refers to the j microblog post of the ith event. The goal to be achieved is to judge whether* $E_{i,j}$ *are rumors.*

The proposed framework of rumor detection is shown in Figure 1. Firstly, the microblog data set was obtained, which was divided into rumor data and non-rumor data. Additionally, the data set was randomly scrambled to ensure that the distribution of the training set, verification set and test set was roughly the same. For each event in the scrambled data set, there are several forwarding data and comment data. Therefore, we sort them according to time series and then they were divided into 20 groups. For each group of event data after grouping, it is regarded as a paragraph text. Additionally, then the Doc2Vec vector operation is carried out to vectorize text. Meanwhile, the sentiment polarity analysis method is used to analyze the sentiment polarity of the texts in each group and the corresponding sentiment tendency features are obtained. Finally, the sentiment tendency features are combined with the grouped paragraph vectors. The 20 sets of vector matrices were used as the input of the convolutional neural network model with the added attention mechanism for training.

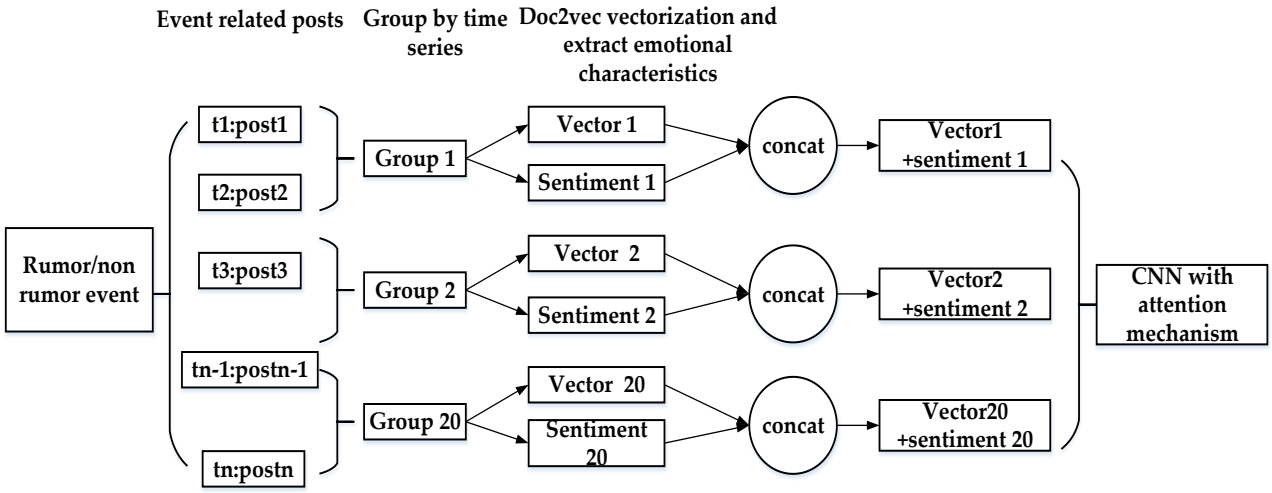

Figure 1. The overall framework of CNN-TS + Sentiment + Attention.

3.2. Construction of Text Vectors

For each group of event data after grouping, it can be regarded as a paragraph text. This paper uses Doc2Vec [34] to encode it as a paragraph vector. Doc2Vec was proposed by Q.Le and T.Mikolov in 2014. It was generated on the basis of the Word2vec model. In which, Word2vec contains two word vector models: Continuous Bag of Words (CBOW) and Skip-Gram [45]. First, the word vector is randomly initialized. Then, the CBOW model predicts the target vocabulary by inputting the context word vector of the target word. Skip-Gram predicts the context word by inputting the target word vector.

The Doc2Vec model with added paragraph attributes is proposed. The model has two prediction forms: PV-DM and PV-DBOW; as shown in Figure 2 PV-DM is similar to CBOW. The difference is that the PV-DM model needs to initialize a paragraph vector randomly. It also needs to predict the target word by averaging or connecting the paragraph vector with the word vector. PV-DBOW predicts context words through the input paragraph vector.

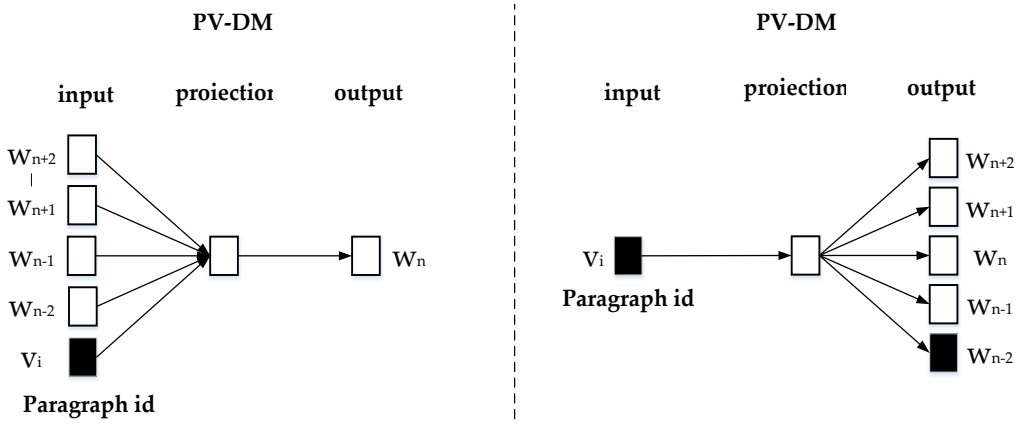

Figure 2. Two training methods of the Doc2Vec model.

The goal of the Doc2Vec model is to maximize the average log probability, the calculation equation is shown in Equation (1).

$$\underset{D,W}{argmax}\frac{1}{N}\sum_{n=k}^{N-k} logp(w_n|w_{n-k,\ldots,}w_{n+k}) \tag{1}$$

In which N is the number of words in the paragraph text, the word vector represented by a word is w_n, the set of word vectors is W, and D is the set of all paragraph vectors. p $(w_n \mid w_{n-k}, \ldots, w_{n+k})$ is used to calculate the maximum likelihood probability of w_n based on the characteristics of $w_{n-k}, \ldots, w_{n+k}$.

Equations (2) and (3) use the Softmax function to predict the probability of the words in the paragraph.

$$p(w_n|w_{n-k},\ldots,w_{n+k}) = \frac{exp(\theta^T x_n)}{\sum_i exp(\theta^T x_i)} \tag{2}$$

$$x_n = h(g_j, w_{n-k},\ldots,w_{n+k}; D, W) \tag{3}$$

In which θ is the parameter of the Softmax function, g_j is a paragraph vector. Additionally, h represents the connection function or the average function. X_n represents the vector composed of k word vectors w_n and paragraph vector g_j. $\theta^T X_n$ is the un-normalized log-probability for each output word n.

3.3. Time Series Grouping and Extraction of Sentimental Characterisitics

For Time Series Grouping operations, J.Wang [10] pointed out that the number of posts for an influential Weibo event is at least a few hundred, or even tens of thousands. Additionally, the number of posts for different events varies greatly. However, the life cycle of a rumor event is divided into the incubation period, the breeding period, the spreading period and the fading period. The text features of Weibo posts in these four time periods and between the four time periods have similar features and changing trends. Therefore, these adjacent posts arranged in time series are regarded as a group, representing a specific life cycle of the event. The consideration for this is: pay more attention to the feature relationship of the content of Weibo posts in each time period so that it can extract the features of the Weibo text between each time period, rather than the feature relationship between a single Weibo post. The development of a Weibo rumor event will have different post content in various time periods. Additionally, this can greatly reduce the complexity of the model input data. The time series segmentation of posts refers to the operation in J. MA [9].

For an event E_i, the start time of the post is $time\_Begin_i$, the end time of the post is $time\_End_i$. Each time $t_{i,j}$ of microblog $m_{i,j}$ is converted to a timestamp between 0 and N, N is the number of time intervals, in this experiment N is taken as 20. N = 20 refers to the experimental setup of Wang et al. [10]. $time\_Stamp_{mi,j}$ is the index of time stamp which m_{ij} falls into, taking the value of 0, 1, ... , N. Then, the calculation Equations for

the $time\_Interval_i$ of the post $time\_Interval_i$ of the event E_i and the $time\_Stamp_{mi,j}$ of the timestamp of each post are shown in Equations (4) and (5).

$$time\_Interval_i = \left\lfloor \frac{time\_End_i - time\_Begin_i}{N} \right\rfloor \tag{4}$$

$$time\_Stamp_{m_{i,j}} = \left\lfloor \frac{t_{i,j} - time\_Begin_i}{time\_Interval_i} \right\rfloor \tag{5}$$

Then, we collect the timestamps of all related Weibo posts and subtract the start timestamp of the corresponding event from all the timestamps of each event. Then, we normalize these timestamps to a 0–1 scale. Finally, the entire timestamp is equally divided into 20 shares in chronological order and the posts in each time window are expressed as shown in Equation (6).

$$T_i = \left(time\_Stamp_{m_{i,j-1}}, time\_Stamp_{m_{i,j}}\right) \qquad j = 1, 2, 3, \ldots, 20 \tag{6}$$

Then, the grouped posts T_i are vectorized with Doc2Vec text and passed into the convolutional neural network training as the input matrix. After the related posts of the i-th Weibo rumor event are grouped, the expression of the vector matrix obtained is shown in Equation (7).

$$V(E_i) = (F_{i,1}, F_{i,2}, \ldots, F_{i,20}) \tag{7}$$

In which E_i is the i-th rumor event and $F_{i,20}$ is the grouped 20 feature vectors.

For extracting sentimental features, we select the SVM model to analyze the polarity of the rumor/non-rumor data set. The overall process is shown in Figure 3. First of all, this paper selects the Weibo data set with sentiment polarity tags. Additionally, then we count the number of positive and negative words in the data set. we use chi-square statistics technology to extract the chi-square features of the text. Then, a Constructing SVM classification is used to obtain the sentiment feature word vector. The chi-square statistic calculation Equations are Equations (8) and (9).

$$\chi^2(w, s) = \frac{N[p(s, w)p(\bar{s}, \overline{w}) - p(\bar{s}, w)p(s, \overline{w})]}{f(s, w)} \tag{8}$$

$$f(s, w) = [p(s, w) + p(s, \overline{w})][p(s, w) + p(\bar{s}, w)] \times [p(\bar{s}, w) + p(\bar{s}, \overline{w})][p(s, \overline{w}) + p(\bar{s}, \overline{w})] \tag{9}$$

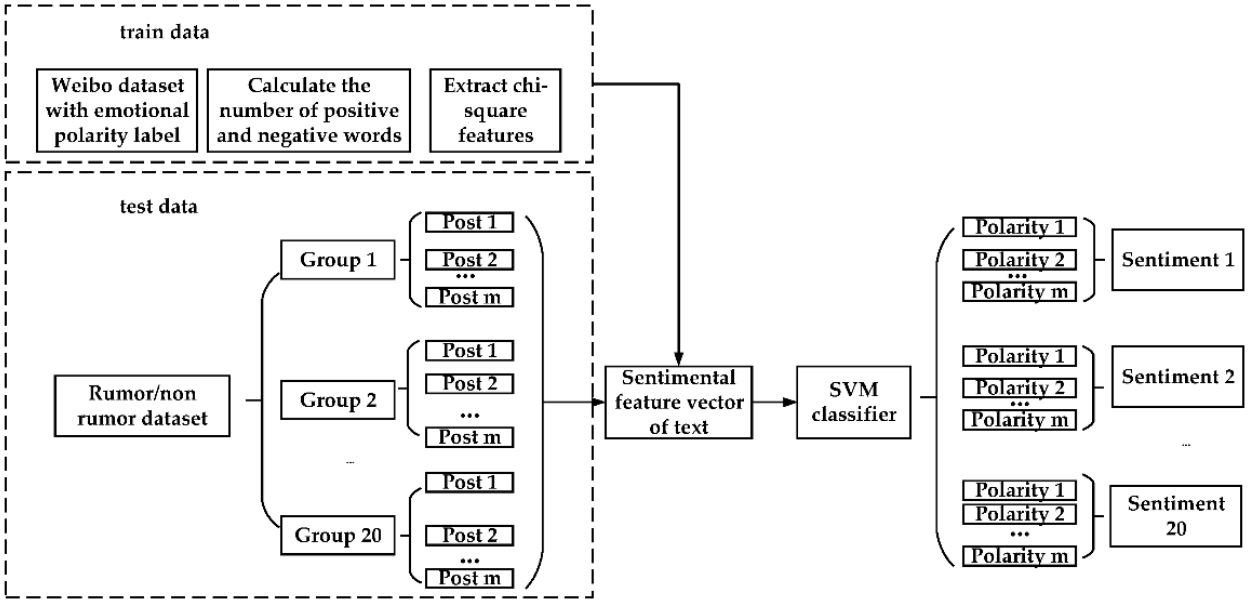

Figure 3. Schematic diagram of the SVM model used to detect emotion polarity.

In which $\chi^2(w,s)$ represents the chi-square measurement of word w in the sentiment category s, N represents the size of the sentiment training data set, $p(s,w)$ represents the number of texts containing the word w in the sentiment category s, $p(\overline{s},w)$ represents the number of texts containing word w that exclude sentiment category s, $p(s,\overline{w})$ represents the number of texts that do not contain word w in emotion category s, and $p(\overline{s},\overline{w})$ represents no sentiment category s and no word w. Using Equations (8) and (9), the chi-square statistics of each word in a certain sentiment category can be calculated. Additionally, a sentiment feature threshold value can be set for each category to make the feature vector more distinguishable. Then, the sentiment feature word is combined as the emotions feature vector.

The obtained sentiment feature vector is input into the SVM model for training and save the model with the best effect.

The test set is the main data set of this research which contains rumor and non-rumor data. After dividing them into 20 groups according to the time series, each group contains m posts. Additionally, each post in each group is subjected to chi-square statistics to obtain the corresponding sentiment feature vector which is used as the input of the SVM model to obtain each post. Then, the sentiment polarity of posts of each group was obtained by the SVM model. The sentiment tendency features of each time series grouping are calculated by the Equation (10).

$$Sentiment\_feature_i = \frac{sum(positive\_words_i)}{m} \quad i = 1,2,3,\ldots,20 \tag{10}$$

In which, i is the group number and m is the number of posts contained in each group. $positive\_words_i$ is the word with positive part of speech in group i. *sum* is function which can calculate the number of positive words. The reason for this operation is: the sentiment polarity of the post is divided into positive, negative and neutral. If the $Sentiment\_feature_i$ value is high, it means that the posts in the group are positive and the number of negative and neutral posts is small; otherwise, they are negative and neutral and the number of positive posts is small and the two correspond to each other.

3.4. The CNN Model with the Attention Mechanism That Combines Sentimental Characterisitics and Time Series

The convolutional neural network model with the attention mechanism is shown in Figure 4. Regarding the input of the CNN with the attention mechanism, the whole rumor event is divided into 20 groups by the time series algorithm. The text vector features and sentiment polarity features in different time series are obtained by the Doc2Vec algorithm and the SVM algorithm, respectively, (see Section 3.4 for details).

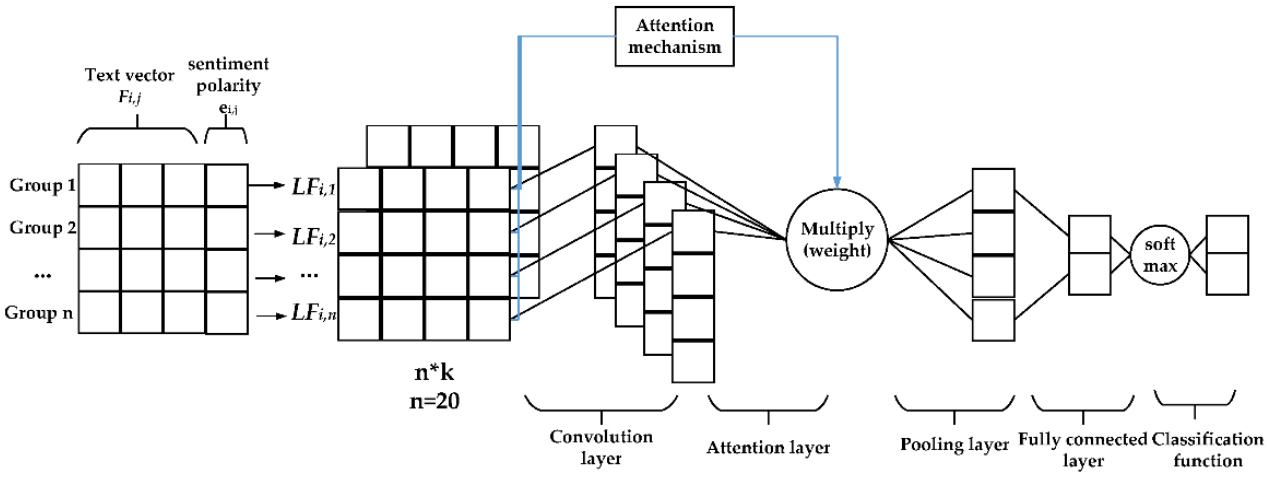

Figure 4. Convolutional neural network model with the attention mechanism.

In terms of the input of CNN model, $LF_{i,j}$ represents the feature vector of the jth group of the rumor event E_i and its dimension is $k + 1$. A feature vector of a microblog rumor event containing n groups of related microblogs is represented, as shown in Equation (11), where n takes 20, and $\oplus$ represents series operation.

$$E_i = LF_{i1} \oplus LF_{i2} \oplus LF_{i3} \oplus \ldots \oplus LF_{in} \tag{11}$$

$LF_{i,j}$ is formed by connecting the text vector and the emotional polarity feature in each group. In this equation, i is the ith event, j is the jth group divided by time series. Its calculation is shown in Equation (12), where n takes 20, $F_{i,j}$ is the text vector in each group, $e_{i,j}$ is the sentiment polarity characteristic, $\oplus$ represents series operation.

Finally, these 20 vectors are used as the input of convolution neural network.

$$LF_{i,j} = F_{i,j} \oplus e_{i,j} \tag{12}$$

Regarding the convolution layer of the CNN with the attention mechanism, we use the convolution kernel filter $w \in R^{h \times k}$ to perform convolution operation on the input matrix to extract new feature vectors. Equation (13) is the operation calculation formula of the three convolution kernels selected by the first hidden layer.

$$\begin{aligned} a_1^{(1)} &= f(W_1^{(1)}F + b_1^{(1)}) \\ a_2^{(1)} &= f(W_2^{(1)}F + b_2^{(1)}) \\ a_3^{(1)} &= f(W_3^{(1)}F + b_3^{(1)}) \end{aligned} \tag{13}$$

In which ${W_1}^{(1)}$, ${W_2}^{(1)}$, ${W_3}^{(1)}$ are the input weights of each convolution kernel and ${b_1}^{(1)}$, ${b_2}^{(1)}$, ${b_3}^{(1)}$ are the biases of each convolution kernel. F is input vector, composed of $F_{i1}, \ldots, F_{in}$. f is a non-linear function; the model used in this paper is the ReLU activation function. $a = [{a_1}^{(1)}, {a_2}^{(1)}, {a_3}^{(1)}]$ is the newly generated feature vector.

Regarding the Attention layer of the CNN with the attention mechanism, we put the attention layer between the convolutional layer and the pooling layer (see Figure 4 for details). Finally, the output after the attention layer and the output after the convolution layer are multiplied as the input of the pooling layer. The input matrix represents the features of the text. The contribution of the regions in the matrix to classification is not equally important. The spatial attention mechanism is used to find the most important part of the input matrix for processing. The calculation formula is shown in Equation (14).

$$\begin{aligned} M(F) &= \sigma(f([AvgPool(F); MaxPool(F)])) \\ &= \sigma(f([F_{avg}; F_{max}])) \end{aligned} \tag{14}$$

In which $M(F)$ represents the weight of the attention operation on the input matrix which is the activation function. This paper uses the Sigmoid function, f is the convolution kernel operation, F is the input matrix, F_{avg} and F_{max} are the global pooling, the average pooling and maximum pooling operations, respectively. *AvgPool* is the average pooling function and *MaxPool* is the maximum pooling function.

Finally, the obtained $M(F)$ is multiplied with the new feature vector obtained by the convolutional layer to emphasize important features. The calculation formula is shown in Equation (15).

$$feature\_vector = M(F) \times a \tag{15}$$

feature_vector is the final input of the pooling layer. $a = [{a_1}^{(1)}, {a_2}^{(1)}, {a_3}^{(1)}]$ is the output of convolution layer mentioned above.

Regarding the pooling layer of the CNN with the attention mechanism, the pooling layer can compress the feature matrix after convolution, so it can make the feature matrix smaller. Additionally, it also can extract the main features. This experiment uses maxpooling to cut the feature matrix into several regions and takes the maximum value. It also can keep the original matrix feature to obtain the pooled feature value.

4. Experiment, Analysis and Discussion

4.1. Experimental Setup

Regarding the datasets of the experiment, we mainly use two datasets. The statistics of the two datasets are shown in Table 1.

Table 1. Statistics of the rumor events dataset and sentiment polarity data.

Data Set	Statistical Type	Statistical Value
Dataset1	Total Comments	119,988
	Total Positive Comments	59,993
	Total Negative Comments	59,995
Dataset2	Involved Users	2,746,818
	Total Posts	3,805,656
	Total Events	4664
	Total Rumor Events	2313
	Total Non-Rumor Events	2351
	Average Posts per Event	816
	Minimum Posts per Event	10
	Maximum Posts per Event	59,138
	Average Time Length per Event	2460.7 h

Dataset1: This experiment used a Weibo data set with sentiment labels collected on the Internet (https://github.com/SophonPlus/ChineseNlpCorpus/blob/master/datasets/weibo_senti_100k/intro.ipynb (accessed on 2 April 2018)). This data set can be used for sentiment, opinion and comment tendency analysis. In this experiment, it is used to train the SVM model to be able to detect the sentiment polarity of Dataset2.

Dataset2: This data set is the public data in the literature of J.Ma [9]. This data set is based on rumor and non-rumors incident obtained from the Sina community management. The center reported various false information on Sina Weibo. In this data set, the original post of the event and all related forwarding/replying information are included.

The Accuracy, Precision, Recall and F1 are usually used to evaluate the effect of the experiment. This paper also selects these four indicators. The specific calculation formula is shown in Equation (16). The evaluating indicators are defined in Table 2.

$$\begin{aligned} accuracy &= \frac{TP+TN}{TP+FP+TN+FN} \\ precision &= \frac{TP}{TP+NP} \\ recall &= \frac{TP}{TP+FN} \\ F1 &= \frac{2*precision*recall}{precision+recall} \end{aligned} \tag{16}$$

The Accuracy reflects the accuracy of the detection method, and the Recall reflects the coverage rate of the detection method. The larger the values of these two indicators are, the better the overall effect of the detection method is.

Table 2. Definition of evaluating indicators.

	Positive	Negative
True	True Positive (TP)	True Negative (TN)
False	False Positive (FP)	False Negative (FN)

We compare and verify the CNN-TS-Sentiment-Attention (CTSA) model proposed in this paper with the Weibo rumor detection model proposed by previous scholars which are based on the traditional ML algorithm DT-Rank (2015), SVM-TS (2015), LK-RBF (2016), $SVM_{com}{}^{DTS}$ (2018), $SVM_{all}{}^{DTS}$ (2018) and the DNN-based methods of GRU-2 (2016), CNN (2017), CallAtRumors (2018), CNN-TS (2021), CNN-TS^{SVM} (2021).

DT-Rank [46]: Z. Zhao proposed a technique based on search query phrases and clustering similar posts together. Then, Z. Zhao collected related posts that do not contain these simple phrases. This model has achieved good performance for rumor detection.

SVM-TS [18]: J. Ma used time series to capture the changes in social background features of rumors over time and then used SVM algorithm to detect rumors.

LK-RBF [47]: J. Sampson proposed two methods based on label and network link dialogue to solve the problem of implicit data. This paper chooses the link-based method with Radial Basic Function (RBF) which achieves the best results in rumor detection.

$SVM_{com}{}^{DTS}$ and $SVM_{all}{}^{DTS}$ [19]: Z-H. Wang constructed the $SVM_{com}{}^{DTS}$ model to capture the dynamic sequence features that change over time. Additionally, on this basis, Z-H. Wang proposed a new rumor detection model $SVM_{all}{}^{DTS}$ with added post popularity, post ambiguity of the spread of rumor events and post spread. This paper compares these two models with our proposed model.

GRU-2 [9]: The author uses the RNN model for rumor detection for the first time. In Jing Ma's work, five models of tanh-RNN, LSTM-1/LSTM-2 and GRU-1/GRU-2 were implemented. This paper selects the best GRU-2 model for comparison and verification.

CNN [29]: Z. Liu proposed to use the CNN model to detect rumors. The scholar used Doc2Vec to make the post text vector and set the dimension of the filter to the length of the text vector to achieve a good detection effect.

CallAtRumors [48]: T. Chen proposed a deep attention model based on a recurrent neural network. This model can selectively learn the time series representation of posts, collect different features with specific focus and mine hidden features. This model is better than the baseline model in performance.

CNN-TS and $CNN\text{-}TS^{SVM}$ [10]: J. Wang merged time series features on the basis of the CNN model and proposed the CNN-TS model. The scholar also improved the classification function, enhanced the generalization ability and proposed a new $CNN\text{-}TS^{SVM}$ model. This paper selects these two models for comparison.

Regarding the setup of the CNN with the attention mechanism, the CNN-TS model proposed by J. Wang et al. [10] is composed of an input layer, a convolution layer, a pooling layer and a fully connected layer. Additionally, they proved that the model can achieve the best recognition effect. The model proposed in this paper is improved on the basis of J. Wang. We refer to the settings of the J.Wang's model to set up our model. The details are shown in Table 3.

Table 3. Setup of the CNN with the attention mechanism.

Layer	Setup
Convolution Layer	One layer; Conv2d(1, 3, (3, 51))
Attention Layer	One layer; Conv2d(2, 3, (3, 51), padding = 0, bias = Flase); Sigmoid()
Pooling Layer	One layer; MaxPool2d((2, 1), 1)
Fully Connected Layer	Linear(51, 4)
Classify Function	Softmax

The height of the filter is set to 3, the number is 3, the vector dimension k of the text is set to 50, the size of the filter is also set to 50 and the dropout rate is 0.5. And 10-fold cross validation is used for the experiment.

4.2. Experimental Results and Discussion

First, we analyze evaluation measures of compared methods. It can be concluded that the effect of the DT-Rank model is poor by analyzing Table 4, because the regular expression used by the model only accounts for 1.6% of Weibo. LK-RBF and SVM-TS extract more microblog posts with more features, so the effect is better. Compared with the

DT-Rank algorithm, the accuracy of the LK-RBF model is increased by 3.3%, the accuracy is increased by 3.7% and the recall is improved by 4.0%. The SVM-TS model improves more on these evaluation indicators. This is because the SVM-TS model not only retains the good generalization ability of SVM, but also consider these features that change over time. Similarly, the $SVM_{com}{}^{DTS}$ model uses a dynamic time series algorithm to capture characteristic variables that changed over time. Compared with SVM-TS, the accuracy, precision, recall and F1 are increased by 3.6%, 3.7%, 3.7% and 3.7%, respectively, which proves that the model that considers the time series is more effective. In addition, the $SVM_{all}{}^{DTS}$ model obtained by adding three new features (post popularity, post ambiguity and post spread) shows a roughly 1.7% improvement in evaluation indicators compared with $SVM_{com}{}^{DTS}$.

Table 4. Performance of rumor event detection methods on Sina Weibo.

Method		Accuracy	Precision	Recall	F1
Traditional ML-based	DT-Rank(2015)	0.648	0.649	0.649	0.648
	LK-RBF(2016)	0.681	0.686	0.689	0.626
	SVM-TS(2015)	0.796	0.797	0.796	0.796
	$SVM_{com}{}^{DTS}$(2018)	0.832	0.834	0.833	0.833
	$SVM_{all}{}^{DTS}$(2018)	0.849	0.850	0.849	0.849
DNN-based	GRU-2(2016)	0.833	0.833	0.834	0.833
	CNN(2017)	0.844	0.847	0.844	0.844
	CallAtRumors(2018)	0.866	0.867	0.866	0.867
	CNN-TS(2021)	0.851	0.839	0.866	0.852
	$CNN\text{-}TS^{SVM}$(2021)	0.866	0.868	0.860	0.863
	CNN-TS + Sentiment(ours)	0.862	0.834	0.903	0.867
	CNN-TS + Sentiment + Attention(ours)	0.867	0.833	0.916	0.872
	CNN-TSSVM + Sentiment + Attention(ours)	0.882	0.869	0.894	0.882

Compared with the traditional machine learning methods above, when the time series features of rumor events are also extracted by the time series algorithm, the rumor recognition effect of model with time series features is stronger than the model without time series features. This proves that adding time series features to the model can effectively detect rumors.

The rumor recognition effect of the CNN-TS + Sentiment model is better than the abovementioned traditional machine learning methods. Compared with the DT-Rank, LK-RBF, the accuracy, precision, recall and F1 values of the model have been improved by more than 18%. Although, compared with $SVM_{all}{}^{DTS}$, the model increases the accuracy, recall and F1 values by 1.3%, 5.4%, 1.8%, respectively. Although the improvement is relatively small, our model greatly reduces the input parameters and reduces the number of model training iterations. The input size of $SVM_{all}{}^{DTS}$ is also $50 \times k$, but the input size of our model is $20 \times (k + 1)$, and k is the dimension of input vector. Additionally, $SVM_{all}{}^{DTS}$ needs to manually extract topic features, user features and content features changed over time. It consumes a lot of manpower and time. Therefore, the effect of our model can be considered significant.

The CNN-TS + Sentiment model proposed in this paper is based on the CNN-TS model with time-varying sentiment tendency features which increase the accuracy, recall

and F1 values by 1.1%, 3.7% and 1.5%, respectively. This proves that considering changes in sentiment polarity features in different life cycles (incubation period, breeding period, spreading period and fading period) is effective for rumor detection. In addition, the attention mechanism is added on this basis to obtain different importance levels (weights) of the input feature matrix and the CNN-TS + Sentiment + Attention model is obtained. The accuracy rate and other evaluation indicators are roughly increased by 1.5~5%. Compared with the CNN-TS model, CNN-TS + Sentiment has improved the accuracy, recall and F1 values. It is proven that the sentiment polarity features in different time series have a positive effect on rumor recognition. Moreover, the input size of the CNN-TS + Sentiment model is 20 × 51 and the input size of the CNN-TS model is 20 × 50. Only by adding 20 input data can the accuracy and other indicators be increased by more than 1.5 %, which is feasible. Similarly, the CNN-TS + Sentiment + Attention model is compared with the CNN-TS model and the CNN-TS + Sentiment model. The four evaluation indexes are also improved. This proves that adding a simple spatial attention mechanism can enhance the input features and improve the rumor recognition effect. In order to fully prove the positive effect of sentiment polarity features and the attention mechanism in different time series on rumor recognition, we apply these two operations to the CNN-TSSVM model. Compared with the CNN-TSSVM model, accuracy, precision, recall and F1 value are improved by more than 2%. The experimental results fully prove that the application of sentiment polarity features and the attention mechanism in the CNN and TS algorithms can effectively identify rumors.

Second, we analyze the parameters of the compared methods. It can be seen from Figure 5 that the input of the CNN [29] is all the posts of a rumor or non-rumor event. If there are n posts and the encoding dimension of the text vector is k, then the input size of the CNN model is $n \times k$. The input of CNN-TS [10] is the post grouped by the time series algorithm. n posts are divided into m groups and the coding dimension of the text vector is still k, so the input size of the CNN-TS model is $m \times k$. This greatly reduces the input parameters. Our CNN-TS + sentiment + Attention model is improved on the basis of the CNN-TS model and n posts are still divided into m groups. The feature coding of posts has a k-dimensional text vector and a sentiment polarity feature. Then, the vector feature dimension of posts is $k + 1$ and the input dimension of CNN-TS + sentiment + Attention is $m \times (k + 1)$. Compared with CNN-TS, only m data are added, but the rumor recognition effect is enhanced. The details are shown in Table 5.

Table 5. Comparison of parameters of the rumor detection models.

Model	Parameter
CNN	59,138 × 50
CNN-TS	20 × 50
CNN-TS + Sentiment	20 × 51
CNN-TS + Sentiment + Attention	20 ×51

CNN-TS + sentiment + Attention is less than the CNN from 59,138 × 50 to 20 × 51, which can reduce the calculation of the model and accelerate the training speed of the model. In addition, although the input parameters are reduced, the experimental results show that the rumor recognition effect is increased, indicating that the packet processing of time series is effective, and the compressed input can still express the rumor features well. With the input of CNN-TS, although 20 data are added, our model is superior to the CNN-TS model in the four important evaluation indexes of rumor recognition. Additionally, this small amount of parameter increase is almost negligible. Reducing the input parameters can greatly reduce the amount of calculation of the model and reduce the number of iterations of model training. Therefore, the training speed of the model is accelerated.

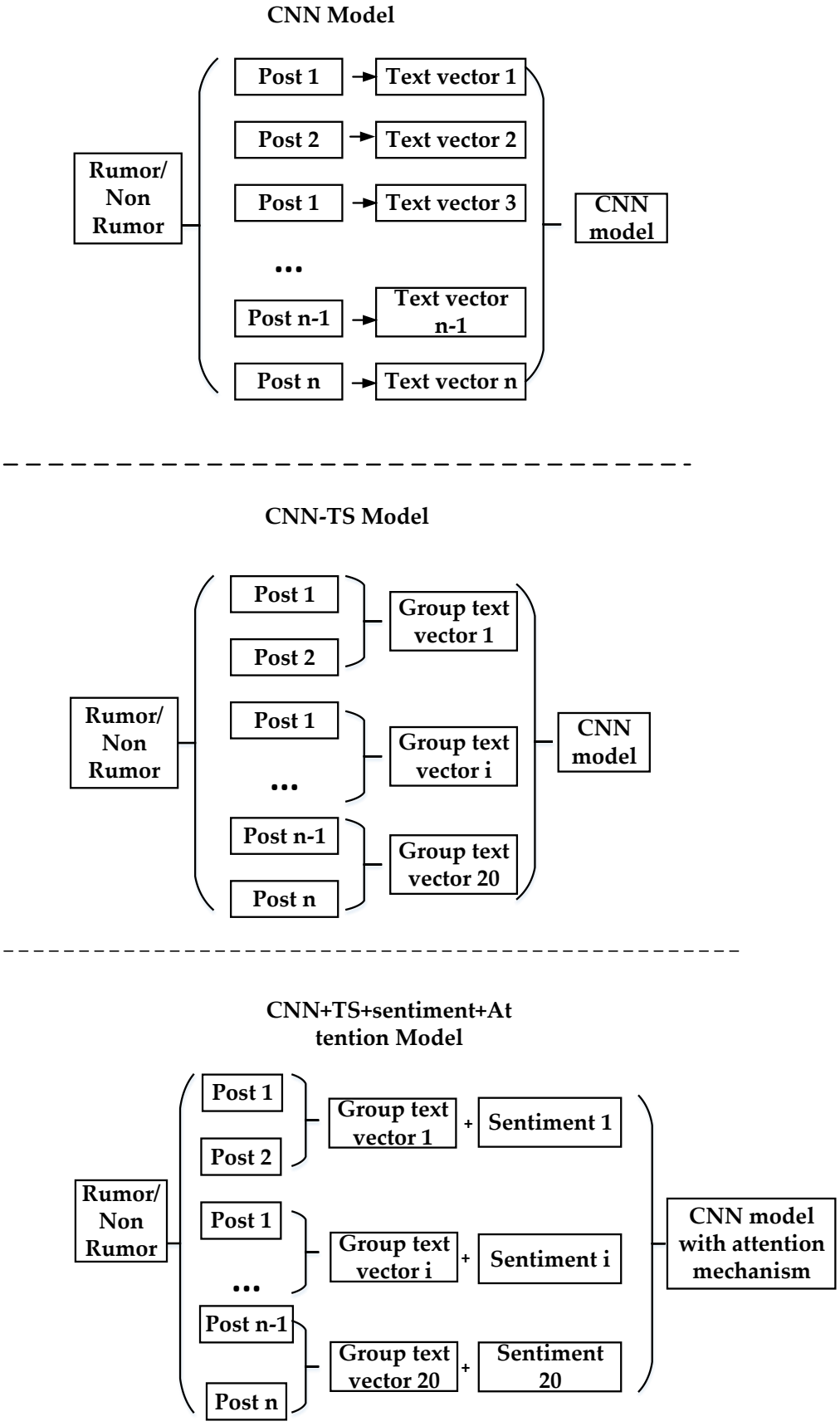

Figure 5. Input comparison of the CNN, CNN-TS, CNN-TS + Sentiment + Attention.

Last, we analyze on epochs of compared methods. The loss of the model is the difference between the predicted value and the real value which can be used to measure the quality of a model. In this experiment, we use 10-fold cross validation to train and test the model. In order to compare the effects of iterations in CNN-TS, CNN-TS + Sentiment and CNN-TS + Sentiment + Attention on the model, we select the first-fold test loss value. See Figure 6 for details.

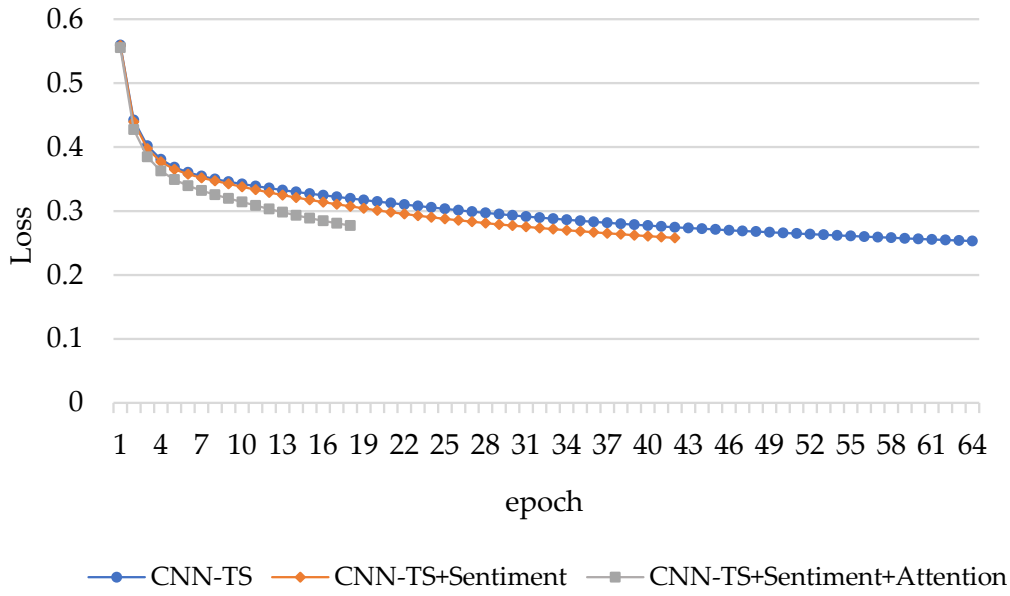

Figure 6. Iteration times and losses of CNN-TS, CNN-TS + Sentiment, CNN-TS + Sentiment + Attention.

It can be seen from Figure 6 that the loss value of CNN-TS model proposed by J. Wang is reduced to 0.2530 after 64 iterations under the first fold data. In order to prevent overfitting, the model takes an early-stopping operation. The CNN-TS + Sentiment proposed in this paper only needs 42 iterations and the loss value reaches stability and stops training. The improved CNN-TS + Sentiment + Attention is more optimized. It only needs 18 iterations, after which the model stops training. Additionally, the loss value remains unchanged. This verifies the superiority of our model in training speed.

5. Final Remarks

The CNN model has a good effect in the recognition of rumors. However, existing CNNs do not comprehensively consider time series characteristics, emotional polarity and parameter reduction when constructing the input of deep neural networks, which affects the further improvement of the model.

In order to reduce the input size of deep neural network without losing context information and consider the time series features of events, we use the time series algorithm (DTS) to divide the whole development cycle of rumors into groups. The features of context and polarity in different time series can be used to identify rumors. In addition, the input of the neural network model is greatly reduced. In order to emphasize the feature representation of the input, we add a layer of spatial attention mechanism to the convolutional neural network which can adjust the feature weight of the input. As shown by the experiment, the proposed model was better than the latest 10 benchmark models in accuracy, precision, recall and F1. It verifies that the introduced sentiment polarity and attention mechanism have a positive effect in the CNN on rumor recognition.

Author Contributions: Conceptualization, data curation, validation Y.P.; methodology, software J.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the National Natural Science Foundation of China (61966017, 61662032); the Humanities and Social Sciences Project of Jiangxi Province (JC19121).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The datasets presented in this study are available on https://github.com/SophonPlus/ChineseNlpCorpus/blob/master/datasets/weibo_senti_100k/intro.ipynb (accessed on 2 April 2018), https://github.com/majingCUHK/Rumor_RvNN (accessed on 2 April 2018) and [14,18].

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Tan, X.; Zhuang, M.; Lu, X.; Mao, T. An Analysis of the sentiment Evolution of Large-Scale Internet Public Opinion Events Based on the BERT-LDA Hybrid Model. *IEEE Access* **2021**, *9*, 15860–15871. [CrossRef]
2. He, G.; Lu, X.; Li, Z. Automatic Rumor Identification on Microblog. *Libr. Inf. Serv.* **2021**, *57*, 114–120.
3. Yang, F.; Liu, Y.; Yu, X. Automatic detection of rumor on sina weibo. In Proceedings of the 2012 the ACM SIGKDD Workshop on Mining Data Semantic, Beijing, China, 12–16 August 2012.
4. Sicilia, R.; Giudice, S.; Pei, Y. Twitter rumour detection in the health domain. *Expert Syst. Appl.* **2021**, *110*, 33–40. [CrossRef]
5. Qazvinian, V.; Rosengren, E.; Radev, D.R. Rumor has it: Identifying misinformation in microblogs. In Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP), Edinburgh, UK, 27–31 July 2011.
6. Gan, W.; Zhang, H.; Cheng, R. A Study on Online Detection of micro-blog Rumors Based on Naive Bayes Algorithm. In Proceedings of the 2020 Asia-Pacific Conference on Image Processing, Electronics and Computers (IPEC), Dalian, China, 14–16 April 2020.
7. Mao, E.; Chen, G.; Liu, X.; Wang, B. Research on detecting microblog rumors based on deep features and ensemble classififier. *Appl. Res. Comput.* **2016**, *33*, 3369–3373.
8. Wang, Z.; Guo, Y.; Wang, J. Rumor Events Detection from Chinese Microblogs via Sentiments Enhancement. *IEEE Access* **2019**, *7*, 103000–103018. [CrossRef]
9. Ma, J.; Gao, W.; Mitra, P.; Kwon, S.; Jansen, B.J.; Wong, K.F.; Cha, M. Detecting rumors from microblogs with recurrent neural networks. In Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI' 16), New York, NY, USA, 9–15 July 2016.
10. Wang, J.; Peng, Y.; Yu, C. Network Rumor Detection Combining Time Series and Convolutional Neural Network. *J. Chin. Comput. Syst.* 2020. Available online: http://kns.cnki.net/kcms/detail/21.1106.TP.20210420.1121.025.html (accessed on 20 April 2021).
11. Takahashi, T.; Igata, N. Rumor detection on twitter. In Proceedings of the 6th International Conference on Soft Computing and Intelligent Systems and 13th International Symposium on Advanced Intelligence Systems (SCIS-ISIS), Kobe, Japan, 23–27 April 2012.
12. Kumar, K.K.; Geethakumari, G. Detecting misinformation in online social networks using cognitive psychology. *Hum. Cent. Comput. Inf. Sci.* **2014**, *9*, 4–14. [CrossRef]
13. Yao, X.; Liang, G.; Gu, C. Rumors clarification with minimum credibility in social networks. *Comput. Netw.* **2021**, *193*, 108123. [CrossRef]
14. Bingol, H.; Alatas, B. Rumor Detection in Social Media using machine learning methods. In Proceedings of the 2019 1st International Informatics and Software Engineering Conference (UBMYK), Ankara, Turkey, 1–5 November 2019.
15. Luo, S.; Wang, J.; Li, B. Study on Rumor Discrimination Based on Improved Combinatorial Optimization Decision Tree. *Comput. Simul.* **2018**, *35*, 219–223.
16. Cai, G.; Bi, M.; Liu, J. A novel rumor detection method based on features of labeled cascade propagation tree. *Comput. Sci. Eng.* **2018**, *40*, 1488–1495.
17. Zeng, Z.; Wang, J. Research on Microblog Rumor Identification Based on LDA and Random Forest. *J. China Soc. Sci. Tech. Inf.* **2019**, *38*, 89–96.
18. Ma, J.; Gao, W.; Wei, Z.; Lu, Y.; Wong, K.F. Detect rumors using time series of social context information on microblogging websites. In Proceedings of the 24th ACM International Conference on Information and Knowledge Management (CIKM), Melbourne, SA, Australia, 19–23 October 2015.
19. Wang, Z.-H.; Guo, Y. Automatic rumor events detection on Chinese microblogs. *J. Chin. Inf. Process.* **2019**, *33*, 132–140.
20. Xu, Y.; Wang, C.; Dan, Z. Deep Recurrent Neural Network and Data Filtering for Rumor Detection on Sina Weibo. *Symmetry* **2019**, *11*, 1408. [CrossRef]
21. Ruchansky, N.; Seo, S.; Liu, Y. Csi: A hybrid deep model for fake news detection. In Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (CIKM'17), Singapore, 6–10 November 2017.

22. Wu, Z.; Pi, D.; Chen, J. Rumor detection based on propagation graph neural network with attention mechanism. *Expert Syst. Appl.* **2020**, *158*, 113595. [CrossRef] [PubMed]
23. Yin, P.; Pan, W.; Peng, C. Research on Early Detection of Weibo Rumors Based on User Characteristics Analysis. *J. Intell.* **2020**, *39*, 81–86.
24. Tarnpradab, S.; Hua, K.A. Attention Based Neural Architecture for Rumor Detection with Author Context Awareness. In Proceedings of the 2018 13th International Conference on Digital Information Management (ICDIM), Berlin, Germany, 8–11 August 2018.
25. Li, J.; Ni, S.; Kao, H.Y. Birds of a Feather Rumor Together? Exploring Homogeneity and Conversation Structure in Social Media for Rumor Detection. *IEEE Access* **2020**, *8*, 212865–212875. [CrossRef]
26. Bian, T.; Xiao, X.; Xu, T.; Zhao, P.; Huang, W. Rumor detection on social media with bi-directional graph convolutional networks. In Proceedings of the 2020 34th Association for the Advance of Artificial Intelligence (AAAI), New York, NY, USA, 7–12 February 2020.
27. Ma, J.; Gao, W.; Wong, K.F. Rumor detection on twitter with tree-structured recursive neural networks. In Proceedings of the 2018 56th Annual Meeting of the Association for Computational Linguistics (ACL), Melbourne, Australia, 26–28 March 2018.
28. Kumar, S.; Carley, K. Tree LSTMs with Convolution Units to Predict Stance and Rumor Veracity in Social Media Conversations. In Proceedings of the 2019 57th Annual Meeting of the Association for Computational Linguistics (ACL), Florence, Italy, 27 July–2 August 2019.
29. Liu, Z.; Wei, Z.H.; Zhang, R.X. Rumor detection based on convolutional neural network. *J. Comput. Appl.* **2017**, *37*, 3053–3056.
30. Santhoshkumar, S.; Badu, L. Earlier detection of rumors in online social networks using certainty-factor-based convolutional neural networks. *Soc. Netw. Anal. Min.* **2020**, *10*, 10–20. [CrossRef]
31. Mi, Y.; Tang, H. Rumor Identification Research Based on Graph Convolutional Network. *Comput. Eng. Appl.* **2020**, *57*, 161–167.
32. Yu, K.; Jiang, H.; Li, T.; Han, S.; Wu, X. Data Fusion Oriented Graph Convolution Network Model for Rumor Detection. *IEEE Trans. Netw. Serv. Manag.* **2020**, *17*, 2171–2181. [CrossRef]
33. Bai, N.; Meng, F.; Rui, X.; Wang, Z. Rumour Detection Based on Graph Convolutional Neural Net. *IEEE Access* **2021**, *9*, 21686–21693. [CrossRef]
34. Quoc, L.; Mikolov, T. Distributed representations of sentences and documents. In Proceedings of the 31st International Conference on International Conference on Machine Learning (ICML'14), Beijing, China, 21–26 June 2014.
35. Bharti, M.; Jindal, H. Automatic Rumour Detection Model on Social Media. In Proceedings of the 2020 Sixth International Conference on Parallel, Distributed and Grid Computing (PDGC), Waknaghat, India, 6–8 November 2020; pp. 367–371.
36. Zu, K.; Zhao, M.; Guo, K. Research on the detection of rumor on Sina Weibo. *J. Chin. Inf. Process.* **2017**, *31*, 198–204.
37. Wang, Z.; Guo, Y. Empower rumor events detection from Chinese microblogs with multi-type individual information. *Knowl. Inf. Syst.* **2020**, *62*, 3585–3614. [CrossRef]
38. Wang, Z.; Guo, Y. Rumor events detection enhanced by encoding sentimental information into time series division and word representations. *Neurocomputing* **2020**, *397*, 224–243. [CrossRef]
39. Lu, Y.J.; Li, C.T. GCAN Graph-aware co-attention networks for explainable fake news detection on social media. In Proceedings of the 2020 the 58th Annual Meeting of the Association for Computational Linguistics (ACL), Washington, DC, USA, 5–10 July 2020.
40. Luo, Y.; Ma, J.; Yeo, C. BCMM:a novel post-based augmentation representation for early rumour detection on social media. *Pattern Recognit.* **2021**, *113*, 107818. [CrossRef]
41. Adel, H.; Schutze, H. Exploring diferent dimensions of attention for uncertainty detection. In Proceedings of the 15th conference of the European chapter of the association for computational linguistics (EACL), Valencia, Spain, 3–7 April 2017.
42. Yuan, Y.; Wang, Y.; Liu, K. Perceiving more truth: A dilated-block-based convolutional network for rumor identification. *Inf. Sci.* **2021**, *569*, 746–765. [CrossRef]
43. Zhang, H.; Qian, S.; Fang, Q.; Xu, C. Multi-modal Meta Multi-Task Learning for Social Media Rumor Detection. *IEEE Trans. Multimed.* **2021**, *99*, 1.
44. Li, Q.Z.; Zhang, Q.; Si, L. Rumor Detection by exploiting user credibility information, attention and multi-task learning. In Proceedings of the 2019 57th Annual Meeting of the Association for Computational Linguistics (ACL), Florence, Italy, 27 July–2 August 2019.
45. Mikolov, T.; Chen, K.; Corrado, G. Efficient Estimation of Word Representations in Vector Space. *arXiv* **2013**, arXiv:1301.3781.
46. Zhao, Z.; Resnick, P.; Mei, Q. Enquiring minds: Early detection of rumors in social media from enquiry posts. In Proceedings of the 24th International World Wide Web Conferences Steering Committee, Florence, Italy, 18–22 May 2015.
47. Sampson, J.; Morstatter, F.; Liu, H.; Wu, L. Leveraging the implicit structure within social media for emergent rumor detection. In Proceedings of the 25th ACM International Conference on Information and Knowledge Management, Indianapolis, IN, USA, 24–28 October 2016.
48. Chen, T.; Li, X.; Yin, H.; Wang, Y. *Call Attention to Rumors: Deep Attention Based Recurrent Neural Networks for Early Rumor Detection*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 40–52.