mathematics

MDPI

Article

Rumor Detection in Social Media Based on Multi-Hop Graphs and Differential Time Series

Jianhong Chen, Wenyi Zhang , Hongcai Ma and Shan Yang *

School of Resources and Safety Engineering, Central South University, Changsha 410083, China
* Correspondence: yangshan@csu.edu.cn

Abstract: The widespread dissemination of rumors (fake information) on online social media has had a detrimental impact on public opinion and the social environment. This necessitates the urgent need for efficient rumor detection methods. In recent years, deep learning techniques, including graph neural networks (GNNs) and recurrent neural networks (RNNs), have been employed to capture the spatiotemporal features of rumors. However, existing research has largely overlooked the limitations of traditional GNNs based on message-passing frameworks when dealing with rumor propagation graphs. In fact, due to the issues of excessive smoothing and gradient vanishing, traditional GNNs struggle to capture the interactive information among high-order neighbors when handling deep graphs, such as those in rumor propagation scenarios. Furthermore, previous methods used for learning the temporal features of rumors, whether based on dynamic graphs or time series, have overlooked the importance of differential temporal information. To address the aforementioned issues, this paper proposes a rumor detection model based on multi-hop graphs and differential time series. Specifically, this model consists of two components: the structural feature extraction module and the temporal feature extraction module. The former utilizes a multi-hop graph and the enhanced message passing framework to learn the high-order structural features of rumor propagation graphs. The latter explicitly models the differential time series to learn the temporal features of rumors. Extensive experiments conducted on multiple real-world datasets demonstrate that our proposed model outperforms the previous state-of-the-art methods.

Keywords: rumor detection; multi-hop graph; differential time series; spatiotemporal features

MSC: 68T07

check for updates

Citation: Chen, J.; Zhang, W.; Ma, H.; Yang, S. Rumor Detection in Social Media Based on Multi-Hop Graphs and Differential Time Series. *Mathematics* **2023**, *11*, 3461. https://doi.org/10.3390/math11163461

Academic Editors: Fei Hao, Doo-Soon Park, Carson K. Leung, Wei Song and Yu-Wang Chen

Received: 4 July 2023
Revised: 20 July 2023
Accepted: 7 August 2023
Published: 9 August 2023

1. Introduction

Since the advent of the Internet era, online social networks have become an indispensable part of our lives. Platforms such as Twitter, Facebook, and Sina Weibo, which focus on social networking or possess social networking attributes, have become primary channels for people to access and share information on a daily basis. However, the exponential growth of content on social media platforms has been accompanied by a proliferation of rumors (fake information), which has had a detrimental impact on the online social environment [1]. The widespread dissemination of rumors distorts facts, leading individuals toward erroneous positions and thereby undermining the public opinion within social networks and posing a serious threat to society [2].

Detection methods and intervention strategies for rumors on social networking platforms have received considerable attention. Facebook encourages users to actively flag suspicious information, while Sina Weibo has established a dedicated Weibo Community Management Center to handle user reports of fake information. However, these existing approaches rely solely on manual verification which, although typically accurate, is limited in effectiveness due to the complexity of the identification process and the constraints of human resources in practical application. Consequently, an increasing number of researchers have been dedicating efforts to developing algorithms for detecting rumors, with the aim of automatically identifying rumors on the internet and addressing the challenges posed by the overwhelming volume of rumors that surpass the capacity of manual verification.

Early automatic rumor detection methods primarily relied on traditional machine learning. Researchers utilized feature engineering to model information from various dimensions of rumor events, followed by supervised training of classifiers to classify rumors and non-rumors. For instance, the authors of [3] employed decision trees, those for [4] utilized random forests, and the authors of [5,6] employed support vector machines (SVMs). These methods demonstrated certain rumor detection capabilities but heavily depended on feature engineering, thus exhibiting noticeable limitations. In recent years, however, deep neural networks (DNNs) have gained popularity, eliminating the need for intricate feature engineering. By training on raw data alone, DNNs can achieve optimal performance, making them widely applicable in the field of rumor detection. For instance, the authors of [7] employed recurrent neural networks (RNNs) to learn the textual content of rumors, while those for [8] utilized convolutional neural networks (CNNs) to extract textual information. Furthermore, with the development of graph neural networks (GNNs) and RNNs, effective modeling of the spatiotemporal features of rumors has become feasible. For instance, the authors of [9] used a tree-structured recursive neural network for propagation feature extraction, and those for [10,11] employed graph convolutional neural networks (GCNs) for structural feature extraction. The authors of [12] captured text and propagation features using a graph encoder and decoder model, and those for [13] utilized gated recurrent units (GRUs) to extract a rumor's temporal and propagation features. These methods have demonstrated excellent performance in rumor detection tasks.

Despite the effective progress made in previous work, several issues still remain. First, existing methods primarily utilize GNNs based on message-passing frameworks to learn the structural features of rumors. However, the propagation of rumors follows a tree structure that unfolds based on time and interaction relationships, with the information source serving as the root node. The connectivity within rumor propagation graphs is relatively simple, but the depth exceeds that of typical graphs. Figure 1 illustrates the distinction between typical graphs and rumor propagation graphs. The characteristics of rumor propagation graphs pose limitations on the application of existing GNN frameworks for rumor detection tasks. Specifically, when faced with deeper node relationships, traditional GNNs can only aggregate information from high-order neighboring nodes by stacking multiple layers. However, this approach leads to issues such as oversmoothing of node features and gradient vanishing, resulting in performance degradation of the network [14,15]. Therefore, a challenge lies in how to better extract the interaction relationships among multi-hop neighbors within rumor propagation graphs.

Furthermore, the current approaches for extracting the temporal features of rumors, whether based on dynamic graphs [16–19] or time series [20,21], only focus on learning features at the level of the original semantic information. However, some studies have indicated that word embeddings, which are used to represent semantic information, possess certain distinctive properties. By analyzing arithmetic operations on word embeddings, the authors of [22] discovered that certain word embedding models can encode linguistic relational patterns. Moreover, the authors of [23–25] conducted case studies on the meanings of individual neurons in word embeddings and found systematic distributions of different linguistic attributes within the embeddings. This allows us to consider word embeddings as relatively stable signals and obtain their changing information through differential operation. The advantage of modeling differential time series explicitly is that rumor features can be extracted from a perspective that varies in time series.

To effectively capture the spatiotemporal features of rumors and achieve better detection performance, this paper proposes a novel self-connected multi-hop graph neural network and differential temporal perception (SMGaDTP) model. The model consists of two main components: the self-connected multi-hop graph attention network (SC-MGAT) module based on multi-hop graphs and the differential temporal perception (DTP) module based on differential time series. The former is utilized to capture the structural features of rumors during their propagation, while the latter focuses on extracting features from the temporal aspects of rumors. Additionally, data augmentation techniques such as DropEdge [26] and TemporalDrop are applied to the SC-MGAT and DTP modules, respectively. The main contributions of our work can be summarized as follows:

- This paper proposes a novel SMGaDTP model for rumor detection tasks. Compared with previous works, this model has the capability to simultaneously learn both the deep structural features and temporal features of rumors.
- The SC-MGAT is proposed in this paper, which builds upon the multi-hop graph and incorporates an enhanced message-passing framework to aggregate extensive neighborhood information. Additionally, a self-connected readout mechanism is introduced to achieve hierarchical extraction of global information.
- DTP is proposed in this paper, which models events from the perspective of differential time series to characterize the temporal variations of events. Based on this, a novel local window attention mechanism and GRU are employed to learn temporal features.
- Extensive experiments on real-world datasets demonstrate that the proposed methods outperform the previous state-of-the-art approaches. Further experiments also indicate that the SC-MGAT exhibits a significant improvement over traditional GNNs in addressing the oversmoothing problem.

The outline of this paper is as follows. Section 2 presents an overview of the relevant previous work. Section 3 provides a formalized description of the proposed problem. Section 4 provides a comprehensive introduction to the proposed model. Section 5 conducts extensive experiments to analyze the effectiveness of the model. Section 6 summarizes the findings and discusses the limitations of this paper.

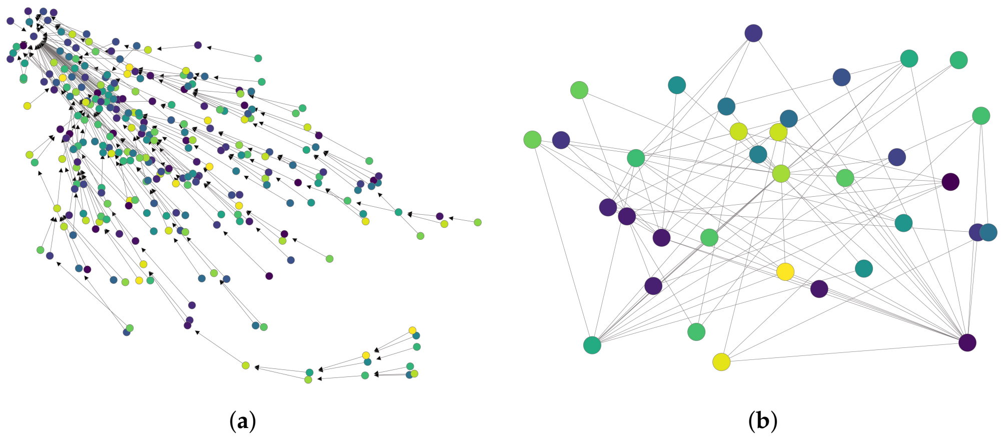

Figure 1. (**a**) A sample from the real-world Weibo dataset representing the propagation process of a specific rumor event. (**b**) A sample from the Les Misérables Co-Occurrence Network dataset, constructed based on Victor Hugo's novel Les Misérables, representing the network graph of character relationships. Rumor propagation graphs often exhibit a significant distance from the root node to the leaf nodes, while typical graphs lack a discernible root node, with all nodes maintaining a high level of connectivity.

2. Related Work

Currently, numerous scholars have proposed various methods for rumor detection tasks employing different feature types and architectures. The main features used include text, visuals, user profiles, statistics, structures, and time sequences. The primary architectures encompass traditional machine learning, CNNs, RNNs, GNNs, and dynamic graph neural networks (DGNNs). In this section, we will primarily focus on reviewing existing work that utilizes text and propagation features. Table 1 shows a summary of the previous work and points out the issues that this paper aims to overcome.

Table 1. Summary of previous work.

Foundation	Method	Limitation
Machine learning	[3,27,28]	Only shallow features can be expressed
Text-based	[7,8,29–37]	Lack of propagation features
RNN-based	[9,13,20,21,38]	Weaker ability to model structural features
GNN-based	[11,39–42]	Lack of temporal and deep structural features
DGNN-based	[16–19]	Lack of deep structural features

2.1. Text-Based Rumor Detection

Text is the core feature of rumors. In the era of traditional machine learning, researchers primarily relied on a series of feature engineering techniques to extract information such as lexical features, symbolic features, and sentiment features from text. For instance, the authors of [3] subdivided text features into string length, presence of emoticons, and personal pronouns. The authors of [27] incorporated the word distribution ratios of rumor and non-rumor information as text features. The authors of [28] included features such as tags, links, and questions present in the text. However, these methods only utilized shallow information and had limited generalization capabilities. Subsequent deep learning algorithms overcame the limitations of traditional machine learning and enabled modeling of deep semantic information. The authors of [7] used RNNs to capture long-range dependencies in text. The authors of [8] employed CNNs to extract deep features from text. The authors of [29] introduced a word-sentence-document structure to extract hierarchical text features while preserving the text's structural hierarchy. Attention mechanisms automatically capture the dependencies between words, giving them a significant advantage over CNNs or RNNs in modeling text content. Consequently, attention mechanisms have been employed to model tweet information in rumors by researchers, such as those in [30,31]. Building upon this, some scholars [32,33] recognized that different domains have distinct linguistic expression forms and incorporated domain-specific terminologies into text features. Additionally, other researchers [34–37] noted the presence of emotional and thematic information in rumor events and extracted features such as sentiment and topics from text for rumor detection tasks.

2.2. Propagation-Based Rumor Detection

In order to effectively capture the multidimensional features of rumors and achieve better detection performance, recent works have focused on exploring the differences between rumors and non-rumors in the propagation process. They have modeled events from the perspective of propagation, including structural and temporal aspects, to achieve more accurate identification. The authors of [38] modeled rumor propagation as a propagation tree and employed kernel learning to extract features from the propagation tree. Similarly, the authors of [9] modeled rumor propagation as a tree and used recursive units to learn propagation features in a top-down and bottom-up manner. The authors of [13] considered the influence of temporal relationships based on the tree structure and proposed a deep spatiotemporal network to simultaneously learn the structural and temporal features of rumors. The authors of [20] combined an RNN with attention mechanisms to capture the contextual changes of semantic information over time in events. The authors of [21] modeled rumors as dynamic time series over time and used GRU units to learn temporal information.

Apart from using RNN architectures to extract propagation features, another mainstream approach is to model events as graphs and utilize graph neural networks under the message-passing framework to learn the propagation features of rumors. For instance, the authors of [39] proposed a GNN-based semi-supervised method for fake news detection. the authors of [11] employed a bidirectional graph convolutional network to learn the propagation and aggregation structures of rumors and included a root node enhancement mechanism in each GCN layer to strengthen the influence of the rumor source on the entire rumor event. The authors of [43] proposed source identification based on graph convolutional networks, using spectral domain convolution to obtain the multi-hop neighbor information of nodes and locate multiple rumor sources without prior knowledge of the underlying propagation model. In addition to the aforementioned methods based on homogeneous graphs, the authors of [40] modeled the global relationships among all source tweets, retweets, and users as a heterogeneous graph to capture richer structural information. The authors of [41] constructed a word-user heterogeneous graph based on the textual content of rumors and the propagation of source tweets, and they proposed a heterogeneous graph attention network framework based on metapaths to capture the global semantic relationships of text content and global structural information of source tweet propagation. The authors of [42] introduced the concept of a joint graph to integrate the propagation structure of all tweets and mitigate sparsity issues, and they utilized network embeddings to learn the representations of nodes in the joint graph.

Considering that static graph structures cannot model the temporal features of rumor propagation, recent research has extended events to dynamic graph structures. The authors of [16] represented rumor posts and their response posts as discrete dynamic graphs and used graph snapshot representation learning with attention mechanisms to capture the structural and temporal information of rumor propagation. The authors of [17] introduced a novel framework for fake news detection based on temporal propagation, modeling the temporal evolution patterns of real-world news as graph evolution patterns under continuous time dynamic diffusion network settings. The authors of [18] modeled each news propagation graph as a series of graph snapshots recorded at discrete time steps and used GCN and attention mechanisms to extract temporal information. The authors of [19] proposed a dual dynamic graph convolutional network that models the dynamic information in message propagation and the dynamic information in the knowledge graph background, learning the two types of structural information in a unified framework.

3. Problem Statement

The propagation of an event in the social network space can be viewed as a set of interacting temporal signals. Therefore, it can be represented by an undirected graph with temporal relationships, denoted as $\mathcal{T} = (\mathcal{V}(t), \mathcal{E}(t))$, where $\mathcal{V}(t) = \{v_1^{(t_1)}, v_2^{(t_2)}, ..., v_n^{(t_n)}\}$ and $\mathcal{E}(t) = \{e_1^{(t_1)}, e_2^{(t_2)}, ..., e_m^{(t_m)}\}$. Here, n represents the number of nodes (i.e., the number of tweets in an event), and m represents the number of edges (i.e., the number of interaction relationships in an event). Each node $v_i^{(t_i)} \in \mathcal{V}(t), i \in [1, n]$ represents a tweet v_i published at time t_i, and each edge $e_j^{(t_j)} \in \mathcal{E}(t), j \in [1, m]$ represents a response relationship between a tweet v_j that appeared at time t_j and a previous tweet $v_p, p \in [1, j-1]$. It is important to note that the response relationship is undirected, and the nodes and edges are sequences with a temporal order. For each tweet $v_i^{(t_i)}$ in the graph, its initial feature representation can be denoted as $\mathbf{h}_i$, which is obtained through processing the information of the original tweet, thereby forming a set of node features $\mathcal{H} = \{\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_n\}$, where $\mathbf{h}_i \in \mathbb{R}^d$ and d represents the dimensionality of the node embeddings. As for the set of edges in the graph, this is represented by the adjacency matrix $\mathbf{A} = (a_{ij})_{n\times n}$, where

$$a_{ij} = \begin{cases} 1, & if \quad e_{ij} \in \mathcal{E}(t) \\ 0, & otherwise \end{cases} \tag{1}$$

The task of rumor detection aims to accurately identify which information in a series of events is a rumor and which is not. For a rumor detection dataset, it can be represented as $\mathcal{C} = \{c_1, c_2, ..., c_N\}$, where c_i represents a specific event in $\mathcal{C}$, N denotes the total number of events in the dataset, and $c_i = (\mathcal{T}_i, \hat{y}_i)$, $\mathcal{T}_i$ is the temporal propagation graph of c_i, while $\hat{y}_i \in \{0, 1\}$ represents the label of c_i (where zero indicates a non-rumor and one indicates a rumor). The objective of this study is to learn a mapping function $\mathcal{F}$ from the dataset $\mathcal{C}$ such that, given the propagation graph $\mathcal{T}_i$ of any other event, we can use $\mathcal{F}$ to track the interactions in $\mathcal{T}_i$ and obtain its predicted label $\hat{y}_i$, enabling accurate classification of the event. In other words, the objective is to achieve $\mathcal{F}(\mathcal{T}_i) \to \hat{y}_i$.

4. Model

4.1. Model Framework

This section provides a brief introduction to the proposed model, and the overall framework of the model is illustrated in Figure 2. The input of the model is a representation of a specific event in the form of a rumor propagation temporal graph $\mathcal{T}$, where each node corresponds to a tweet and each edge represents the replying relationship between tweets. The output of the model is the probability that the event is a rumor.

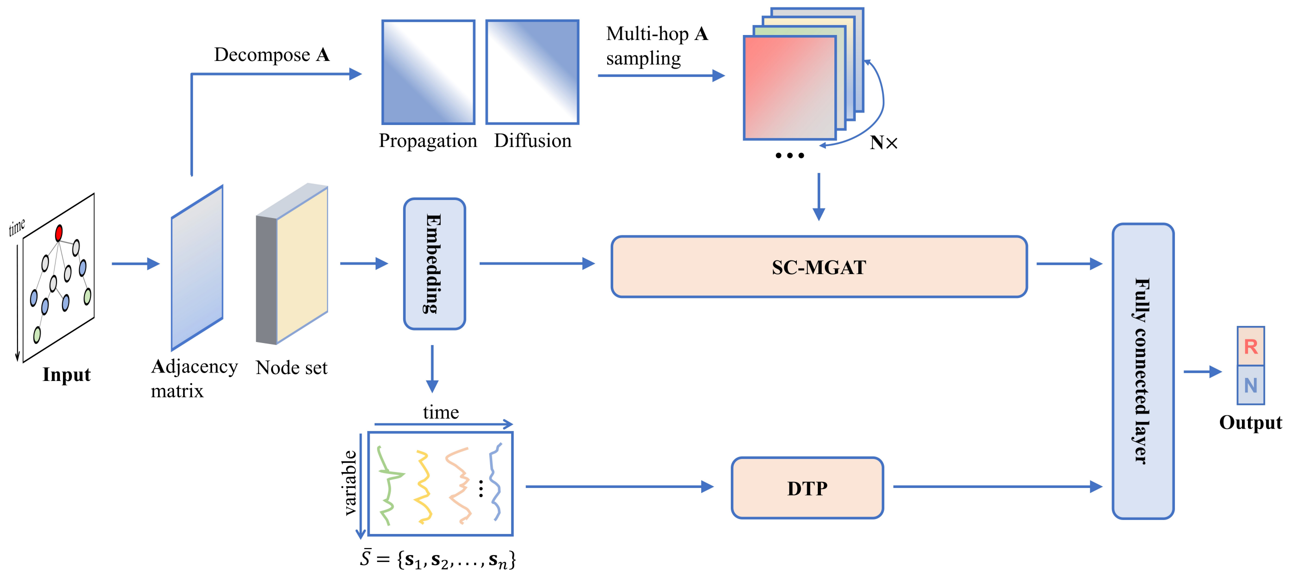

Figure 2. Model framework.

For an input $\mathcal{T}$, we consider two components: the adjacency matrix and the node set. Regarding the adjacency matrix, it is initially decomposed into two directed adjacency matrices representing the paths of propagation and diffusion:

$$(\overleftharpoon{\mathbf{A}}, \overrightharpoon{\mathbf{A}}) = \text{ToDirect}(\mathbf{A}), \tag{2}$$

where ToDirect$(\cdot)$ denotes the decomposition of an undirected adjacency matrix into two distinct directed adjacency matrices and $\overleftharpoon{\mathbf{A}}$ and $\overrightharpoon{\mathbf{A}}$ are the upper triangular and lower triangular matrices, respectively. Subsequently, matrix exponentiation is applied to compute **N** multi-hop adjacency matrices, each containing neighbors at varying distances:

$$\mathbf{A}^{(k)} = \overleftharpoon{\mathbf{A}}^{k} + \overrightharpoon{\mathbf{A}}^{k}, \tag{3}$$

$$\mathbf{A_k} = \text{Filter}(\mathbf{A}^{(k)} = 1), \tag{4}$$

where Filter$(\cdot = 1)$ signifies the preservation of elements in the matrix that are equal to one, $\mathbf{A_k}$ represents the adjacency matrix that exclusively contains k-hop neighbors, $\mathcal{A} = \{\mathbf{A_1}, \mathbf{A_2}, \ldots, \mathbf{A_N}\}$ denotes the collection of multi-hop adjacency matrices, and **N** represents the number of samples taken. As for the node set, the corresponding initial embeddings $\mathcal{H} = \{\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_n\}$ are obtained through an embedding layer, which incorporates information such as the text, timestamp, and structural characteristics of each node.

After obtaining the node embeddings and multi-hop adjacency matrices, we proceed to learn the features of the rumor in terms of both the structural and temporal aspects.

Specifically, the initial node embeddings $\mathcal{H}$ and the collection of multi-hop adjacency matrices $\mathcal{A}$ are fed into the SC-MGAT module for in-depth structural feature learning. On the other hand, the node embeddings $\mathcal{H}$ are sorted in ascending order based on their timestamps:

$$\text{Sorted}(\mathcal{H}) \rightarrow \bar{\mathcal{S}}, \tag{5}$$

and the resulting time series $\bar{\mathcal{S}} = \{\mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_n\}$ is then fed into the DTP module for temporal feature learning. Finally, the structural feature and temporal feature are combined and fed into a classifier composed of linear layers, yielding the ultimate classification result.

4.2. Embedding Layer

The embedding layer is designed to create original feature representations for each input graph. For an input graph $\mathcal{T}$ with n nodes, each node v_i in the graph considers three aspects of information—tweet text, timestamp, and structure—and encodes them separately. Specifically, for the textual content, pretrained word embeddings are utilized to obtain robust text feature representations:

$$\text{WordEmbed}({v_1}^{word}, {v_2}^{word}, ..., {v_n}^{word}) \rightarrow \mathbf{H}_{word}, \tag{6}$$

where $\mathbf{H}_{word} = [\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_n]$, $\mathbf{w}_i \in \mathbb{R}^w$ represents the word embedding, and w is the dimension of embedding. Regarding the timestamp, it is decomposed and encoded to extract information such as the year, month, day, hour, minute, and second:

$$\text{TimeEmbed}({v_1}^{time}, {v_2}^{time}, ..., {v_n}^{time}) \rightarrow \mathbf{H}_{time}, \tag{7}$$

where $\mathbf{H}_{time} = [\mathbf{t}_1, \mathbf{t}_2, ..., \mathbf{t}_n]$, $\mathbf{t}_i \in \mathbb{R}^t$ represents the time embedding and t is the dimension of embedding.

Most existing works have not taken into account encoding the structural information of nodes. However, previous research [44] has demonstrated that the expressive power of the classical GNN is limited by the 1-Weisfeiler-Lehman (1-WL) graph isomorphism test. Furthermore, the authors of [45] highlighted the importance of structural information for graph classification tasks. Encoding structural features for nodes can alleviate these limitations. Additionally, in the propagation process of rumor events, the local neighborhood structure of nodes reflects rich social information and can, to some extent, indicate the influence of nodes throughout the entire event. Therefore, drawing inspiration from [45,46], structural information encoding can be employed for nodes in graph $\mathcal{T}$:

$$\mathbf{H}_{neigh} = [\mathbf{A}^1 \cdot \mathbf{1}, \mathbf{A}^2 \cdot \mathbf{1}, \mathbf{A}^3 \cdot \mathbf{1}, ..., \mathbf{A}^q \cdot \mathbf{1}]^T, \tag{8}$$

$$\mathbf{H}_{identity} = [diag(\mathbf{A}^1), diag(\mathbf{A}^2), diag(\mathbf{A}^3), ..., diag(\mathbf{A}^q)]^T, \tag{9}$$

$$\mathbf{H}_{struct} = \text{Concat}(\mathbf{H}_{neigh}, \mathbf{H}_{identity}), \tag{10}$$

where $\mathbf{A} \in \mathbb{R}^{n \times n}$ represents the adjacency matrix of the graph $\mathcal{T}$, $\mathbf{1} \in \mathbb{R}^n$ represents a vector of ones, $diag(\cdot)$ refers to a vector containing the diagonal elements of a matrix, q represents the number of recursive encodings, $\mathbf{H}_{struct} = [\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_n]$, and $\mathbf{u}_i \in \mathbb{R}^q$ represents the structural embedding.

Finally, the textual, temporal, and structural embeddings are concatenated to obtain the initial embedding for each node:

$$\mathbf{H} = \text{Concat}(\mathbf{H}_{word}, \mathbf{H}_{time}, \mathbf{H}_{struct}), \tag{11}$$

where $\mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_n]$, $\mathbf{h}_i \in \mathbb{R}^d$ represents the node embedding and $d = w + t + q$ is the dimension of embedding.

4.3. Self Connected Multi-Hop Graph Attention Network (SC-MGAT)

A few researchers have noticed the issue of high-order neighbor message passing in GNNs. Inspired by the deep residual network (ResNet) [47], researchers such as those in [15] have employed residual connections to alleviate the problems of gradient vanishing and oversmoothing. However, they are not efficient in aggregating information from multiple hops of neighboring nodes. Another approach proposed in [48] involves concatenating features from multiple hops of neighbors and aggregating them using attention mechanisms. However, this approach simplifies the hierarchical structure of aggregating multi-hop neighbors. Therefore, building upon previous works, the self-connected multi-hop graph attention network (SC-MGAT) is proposed, which consists of two components: the multi-hop graph attention network (Multi-hop GAT) and the self-connected aggregation (SCA). Figure 3 illustrates the workflow of the SC-MGAT.

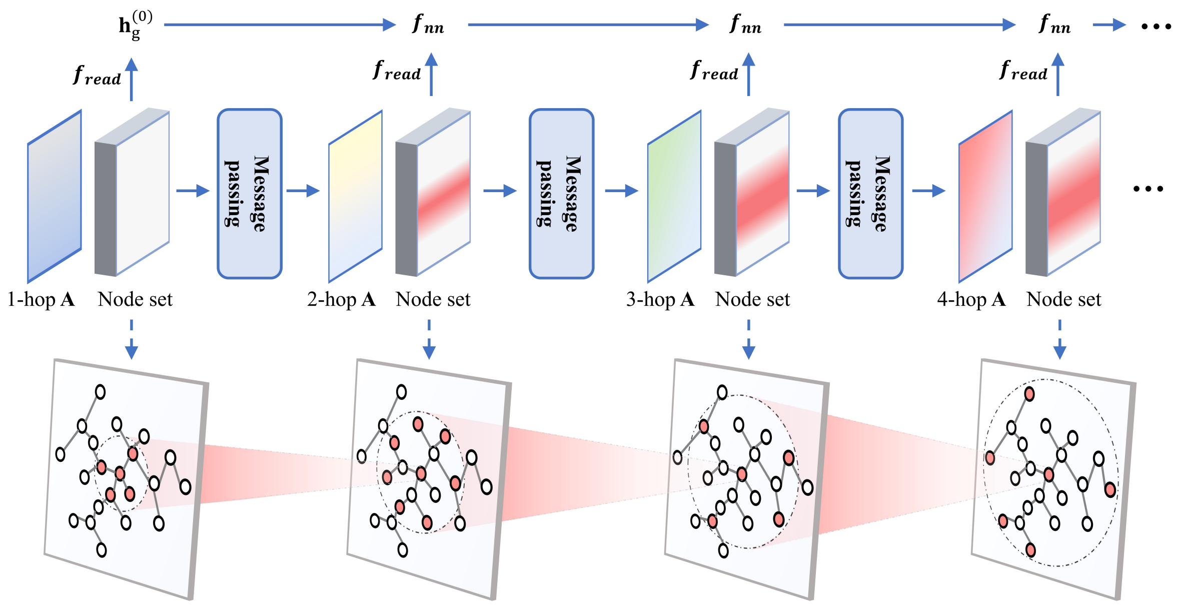

Figure 3. The workflow of the SC-MGAT (taking the example of $\zeta(l) = l$).

4.3.1. Multi-Hop Graph Attention Network (Multi-Hop GAT)

The Multi-hop GAT builds upon and improves the traditional message-passing paradigm. Specifically, for a given graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, the input consists of a set of node representations $\left\{\mathbf{h}_i \in \mathbb{R}^d \mid i \in \mathcal{V}\right\}$ and the corresponding edges $\mathcal{E}$. The output is a new set of node representations $\left\{\mathbf{h}'_i \in \mathbb{R}^{d'} \mid i \in \mathcal{V}\right\}$. The nodes are updated using the following function:

$$\mathbf{h}_v^{(l)} = \phi\left(\mathbf{h}_v^{(l-1)}, f\left(\left\{\mathbf{h}_u^{(l-1)} | u \in \mathcal{N}^{(k)}(v)\right\}\right)\right), \tag{12}$$

This differs from the approach presented in [44], where $\mathcal{N}^{(k)}$ represents the k-hop neighbors of node v, indicating that during the lth layer of message passing, the k-hop neighbors' features from the $l-1$ layer are utilized. The function f denotes the aggregation function, while ϕ represents the update function and

$$k = \zeta(l), \quad \zeta : \mathbb{Z} \geq 1 \to \mathbb{Z} \geq 1, \tag{13}$$

where $\mathbb{Z} \geq 1$ refers to the set of positive integers. For a general GNN, $\zeta(l) \equiv 1$, indicating the continuous use of one-hop neighbors message passing. By selecting different mapping functions ζ, various receptive fields for message passing can be achieved, thus enabling efficient aggregation of global graph information.

Inspired by [49,50], the aggregation and update process of neighbor nodes in this paper is as follows. First, for any node i and its k-hop neighbor j, the scoring function ϑ

is used to compute the edge score between the two nodes. The edge score represents the importance of neighbor j to node i:

$$\vartheta(\mathbf{h}_i, \mathbf{h}_j) = \mathbf{a}^T \text{LeakyReLU}\big(\mathbf{W}_1 \mathbf{h}_i + \mathbf{W}_1 \mathbf{h}_j\big), \tag{14}$$

where $\mathbf{a} \in \mathbb{R}^{d'}$ and $\mathbf{W}_1 \in \mathbb{R}^{d' \times d}$ are trainable parameters. After obtaining the edge scores for all neighbors $j \in \mathcal{N}(i)$, a softmax function is applied to normalize the edge scores. Finally, the node i obtains its new representation $\mathbf{h}'_i$ through weighted aggregation of the edge scores:

$$\partial_{ij} = \text{softmax}_j(\vartheta(\mathbf{h}_i, \mathbf{h}_j)) = \frac{\exp(\vartheta(\mathbf{h}_i, \mathbf{h}_j))}{\sum_{j' \in \mathcal{N}(i)} \exp(\vartheta(\mathbf{h}_i, \mathbf{h}_{j'}))}, \tag{15}$$

$$\mathbf{h}'_i = \sigma\Big(\textstyle\sum_{j \in N(i)} \partial_{ij} \cdot \sigma\big(\mathbf{W}_2 \cdot [\mathbf{W}_1 \mathbf{h}_i \parallel \mathbf{W}_1 \mathbf{h}_j]\big)\Big), \tag{16}$$

where $\mathbf{W}_2 \in \mathbb{R}^{d' \times 2d'}$ is a trainable parameter and σ denotes a nonlinear activation function. Through the aforementioned calculations, the new node feature set after aggregation can be obtained, denoted as $\mathcal{H} = \{\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_n\} \rightarrow \mathcal{H}' = \{\mathbf{h}'_1, \mathbf{h}'_2, ..., \mathbf{h}'_n\}$.

4.3.2. Self-Connected Aggregation (SCA)

For graph-level classification tasks, after updating the features of the graph using a GNN, it is typically necessary to perform a readout operation to extract information that represents the entire graph:

$$\mathbf{h}_g = f_{read}(\mathbf{h}_v | v \in \mathcal{V}), \tag{17}$$

where $\mathbf{h}_g$ represents the global representation of the graph $\mathcal{G}$ and f_{read} refers to the method used to extract the global information from the graph. Common readout methods include global average pooling, Top-K pooling [51], DiffPool [52], and ASAPooling [53]. However, the previous methods only performed the readout operation in the final iteration step, which is not favorable for our proposed Multi-hop GAT.

Specifically, the primary objective of the Multi-hop GAT is to hierarchically aggregate high-order neighbor information to obtain rich graph representations, while performing a readout only in the final layer would result in the loss of information from previous layers. Meanwhile, generating graph-level representations, as pointed out in [54], is equivalent to having a virtual super node in the graph where real nodes aggregate information along virtual edges toward the super node:

$$\mathbf{h}_s \Leftrightarrow \mathbf{h}_g = f_{read}(\mathbf{h}_v | v \in \mathcal{V}), \tag{18}$$

where $\mathbf{h}_s$ represents the representation of the virtual super node. When the readout is performed only in the final layer, the self-loop of the super node is consistently overlooked which, as mentioned in [55], leads to an insufficient representational capacity for the super node.

Therefore, inspired by [54], to better hierarchically aggregate multi-hop neighbor information and enhance the expressive power of global graph representations, we introduce the self-connected aggregation (SCA) module into Multi-hop GAT. In this module, the global information at each layer is determined by both the node information of the current layer and the global information from the previous layer. Specifically, the computation is as follows:

$$\mathbf{h}_g^{(l)} = f_{nn}(\mathbf{h}_g^{(l-1)} + f_{read}(\mathbf{h}_v^{(l)} | v \in \mathcal{V})), \tag{19}$$

where l denotes the layer of the GNN and f_{nn} represents the linear projection. By integrating Multi-hop GAT with SCA, the algorithmic flow of the SC-MGAT can be obtained (see Algorithm 1).

Algorithm 1 SC-MGAT

```
INPUT: Node feature set $\mathcal{H} = \{\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_n\}$, Adjacency set $\mathcal{A} = \{\mathbf{A_1}, \mathbf{A_2}, ..., \mathbf{A_N}\}$
OUTPUT: Final graph representation $\mathbf{h}_g^{(l)}$
 1: $\mathbf{h}_g^{(0)} = f_{read}(\mathbf{h}_v | v \in \mathcal{V})$    //initialization graph representation
 2: for $l = 1 \to \mathrm{M}$ do
 3:     $k = \zeta(l)$
 4:     $\mathbf{A_k} = \mathcal{A}[k]$    //obtain the adjacency matrix containing only the $k$-hop neighbors
 5:     DropEdge($\mathbf{A_k}$)
 6:     for all $\mathbf{h}_v \in \mathcal{H}$ do
 7:         $\mathcal{N}^{(k)}(v) = \mathbf{A_k}[v]$    //obtain the $k$-hop neighbors of node $v$
 8:         $\mathbf{h}_v^{(l)} = \phi\left(\mathbf{h}_v^{(l-1)}, f\left(\left\{\mathbf{h}_u^{(l-1)} | u \in \mathcal{N}^{(k)}(v)\right\}\right)\right)$    //message passing
 9:     end for
10:     $\mathbf{h}_g^{(l)} = f_{nn}\left(\mathbf{h}_g^{(l-1)} + f_{read}\left(\mathbf{h}_v^{(l)} | v \in \mathcal{V}\right)\right)$    //self-connection readout
11: end for
12: return $\mathbf{h}_g^{(l)}$
```

4.4. Differential Temporal Perception (DTP)

Rumor events in social networks evolve in chronological order. When a hot topic emerges, an increasing number of users participate and contribute more information. These pieces of information exhibit rich variations in terms of cycles or trends, such as changes in sentiment polarity and topic shifts. Inspired by [56,57], this paper represents the temporal propagation process of events as a multivariate time series $\bar{\mathcal{S}} = \{\mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_n\}$, where n represents the number of tweets related to a specific event and $\mathbf{s}_i \in \mathbb{R}^d$ denotes the feature representation at each time step. Based on this, the differential temporal perception (DTP) module is proposed to capture the evolutionary features of events at the temporal level. The process of the DTP module is shown in Figure 4.

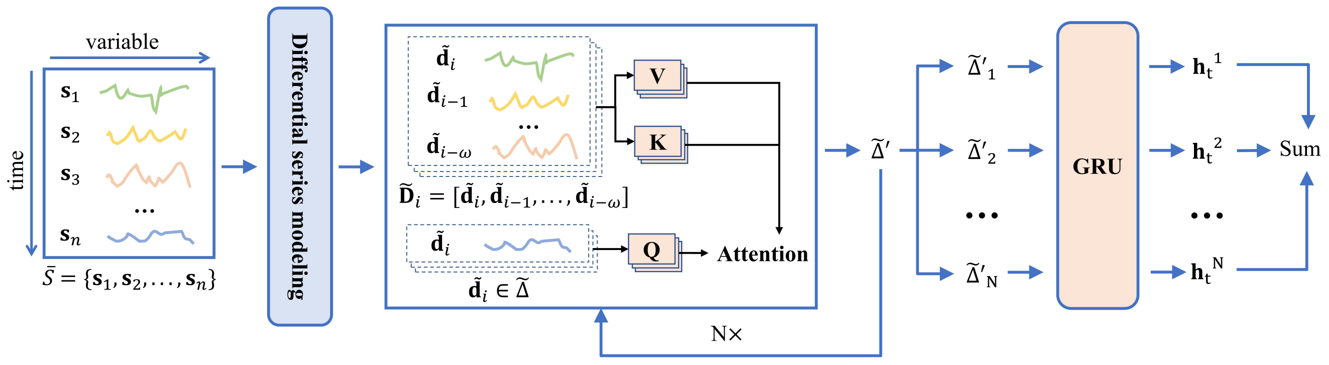
variable
time
$\mathbf{s}_1$
$\mathbf{s}_2$
$\mathbf{s}_3$
...
$\mathbf{s}_n$
$\bar{S} = \{\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_n\}$
Differential series modeling
$\tilde{\mathbf{d}}_i$
$\tilde{\mathbf{d}}_{i-1}$
...
$\tilde{\mathbf{d}}_{i-\omega}$
V
K
$\tilde{\mathbf{D}}_i = [\tilde{\mathbf{d}}_i, \tilde{\mathbf{d}}_{i-1}, \ldots, \tilde{\mathbf{d}}_{i-\omega}]$
$\tilde{\mathbf{d}}_i$
Q
Attention
$\tilde{\mathbf{d}}_i \in \tilde{\Delta}$
N×
$\tilde{\Delta}'$
$\tilde{\Delta}'_1$
$\tilde{\Delta}'_2$
...
$\tilde{\Delta}'_N$
GRU
$\mathbf{h}_t^1$
$\mathbf{h}_t^2$
...
$\mathbf{h}_t^N$
Sum

Figure 4. The workflow of DTP.

The first step of DTP is to perform differential time series modeling. First, to simulate the temporal changes of events, we perform a dropout operation on the initial series $\bar{\mathcal{S}}$ while preserving the temporal relationships:

$$\mathrm{TemporalDrop}(\bar{\mathcal{S}}) \to \mathcal{S}, \tag{20}$$

where $\mathcal{S} = \{\mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_m\}$. Then, the differential time series $\Delta = \{\mathbf{d}_1, \mathbf{d}_2, ..., \mathbf{d}_m\}$ based on the original time series $\mathcal{S}$ is constructed, where $\mathbf{d}_1 = \mathbf{0}$, and $\mathbf{d}_i = \mathbf{s}_i - \mathbf{s}_{i-1}, i \in [2, m]$. Similar to [58], to retain the positional information of the sequence, positional encoding is applied to the differential series, resulting in a series $\mathcal{P} = \{\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_m\}$ with the same dimensions as Δ:

$$PE_{(pos,2i)} = \sin(pos/10000^{2i/c}), \tag{21}$$

$$PE_{(pos,2i+1)} = \cos(pos/10000^{2i/c}), \tag{22}$$

where $pos \in [0, m]$ represents the position in the series and $i \in [0, c]$ represents the dimension. The positional encoding series $\mathcal{P}$ is added to the differential series Δ to obtain a new series $\tilde{\Delta} = \{\tilde{\mathbf{d}}_1, \tilde{\mathbf{d}}_2, ..., \tilde{\mathbf{d}}_m\}$, where $\tilde{\mathbf{d}}_i = \mathbf{d}_i + \mathbf{p}_i$.

Then, the series $\tilde{\Delta}$, enhanced with positional encoding, will undergo local window attention (LWA) calculation. Specifically, to ensure temporal and local dependencies, a fixed window size ω is used, and for any $\tilde{\mathbf{d}}_i \in \tilde{\Delta}$, a corresponding subsequence $\tilde{\mathbf{D}}_i = [\tilde{\mathbf{d}}_i, \tilde{\mathbf{d}}_{i-1}, ..., \tilde{\mathbf{d}}_{i-\omega}]$ is extracted, where $i > \omega$. If there are fewer than ω elements before $\tilde{\mathbf{d}}_i$, then all preceding elements are considered. Subsequently, similar to [58], a multi-head attention calculation is performed for all elements in the subsequence $\tilde{\mathbf{D}}_i$ with corresponding $\tilde{\mathbf{d}}_i$ values:

$$head_p = \text{softmax}(\frac{\mathbf{Q}_p{}^T\mathbf{K}_p}{\sqrt{\varepsilon_p}})\mathbf{V}_p{}^T, \tag{23}$$

$$MultiHead(\tilde{\mathbf{d}}_i, \tilde{\mathbf{D}}_i) = \text{Concat}(head_1, head_2, ..., head_h), \tag{24}$$

Here, $\mathbf{Q}_p = \mathbf{W}_Q{}^p \times \tilde{\mathbf{d}}_i^p$, $\mathbf{K}_p = \mathbf{W}_K{}^p \times \tilde{\mathbf{D}}_i^p$, and $\mathbf{V}_p = \mathbf{W}_V{}^p \times \tilde{\mathbf{D}}_i^p$, where $\mathbf{W}_Q{}^p, \mathbf{W}_K{}^p, \mathbf{W}_V{}^p \in \mathbb{R}^{\varepsilon_p \times \varepsilon_p}$ are trainable weight parameters, $\tilde{\mathbf{d}}_i^p \in \mathbb{R}^{\varepsilon_p}$, and $\tilde{\mathbf{D}}_i^p \in \mathbb{R}^{\varepsilon_p \times (\omega+1)}$. In this paper, we divide the input based on the feature dimension for different heads of the multi-head attention. Hence, $\varepsilon_p = d/h$, where h is the number of heads.

For all $\tilde{\mathbf{d}}_i \in \tilde{\Delta}$, we have $\tilde{\mathbf{d}}'_i = MultiHead(\tilde{\mathbf{d}}_i, \tilde{\mathbf{D}}_i)$. This process yields a new sequence $\tilde{\Delta}' = \{\tilde{\mathbf{d}}'_1, \tilde{\mathbf{d}}'_2, ..., \tilde{\mathbf{d}}'_m\}$ that incorporates local information. The new sequence is then fed into the GRU to learn temporal information. The forward propagation process is as follows:

$$\mathbf{r}_t = \sigma(\mathbf{W}_r \cdot [\tilde{\mathbf{d}}'_t, \mathbf{h}_{t-1}]), \tag{25}$$

$$\mathbf{z}_t = \sigma(\mathbf{W}_z \cdot [\tilde{\mathbf{d}}'_t, \mathbf{h}_{t-1}]), \tag{26}$$

$$\hat{\mathbf{h}}_t = \tanh(\mathbf{W}_h \cdot [\tilde{\mathbf{d}}'_t, (\mathbf{r}_t \odot \mathbf{h}_{t-1})]), \tag{27}$$

$$\mathbf{h}_t = (1 - \mathbf{z}_t) \odot \mathbf{h}_{t-1} + \mathbf{z}_t \odot \hat{\mathbf{h}}_t, \tag{28}$$

where $\mathbf{r}_t$ represents the reset gate, $\mathbf{z}_t$ represents the update gate, and $\mathbf{W}_r, \mathbf{W}_z, \mathbf{W}_h \in \mathbb{R}^{d' \times (d+d')}$ are trainable parameters, while $\tilde{\mathbf{d}}'_t$ represents the input at time t, $\mathbf{h}_{t-1} \in \mathbb{R}^{d'}$ is the hidden state at time $t-1$, and d' is the output dimension of the GRU. The output at the final time step is used as the temporal feature representation of the entire sequence.

Furthermore, by employing a multi-layer stacking approach, it is possible to learn the temporal features over a broader range. Specifically, after obtaining a new sequence $\tilde{\Delta}'$ that fuses local information through LWA, a larger range of information can be further fused on the basis of $\tilde{\Delta}'$:

$$\tilde{\Delta}'_{i+1} = \text{LWA}(\tilde{\Delta}'_i) \tag{29}$$

Stacking LWA enables a linear expansion of the receptive field. Finally, all obtained sequences $\{\tilde{\Delta}'_1, \tilde{\Delta}'_2, ..., \tilde{\Delta}'_{\text{N}}\}$ are processed in parallel using a GRU for temporal feature extraction, and the resulting features are summed together to obtain the final temporal features:

$$\mathbf{h}_t{}^{sum} = \sum_{i=1}^{\mathrm{N}} \mathrm{GRU}(\tilde{\Delta}_i') \tag{30}$$

4.5. Classification Layer

The output of the SC-MGAT module is the structural feature representation $\mathbf{h}_g^{(l)}$, and the output of the DTP module is the temporal feature representation $\mathbf{h}_t{}^{sum}$. These two features are combined through an addition operation to obtain the final feature representation $\mathbf{h}$:

$$\mathbf{h} = \mathbf{h}_g^{(l)} + \mathbf{h}_t{}^{sum} \tag{31}$$

Subsequently, several linear layers followed by nonlinear activation layers are applied to obtain the final predicted label $\hat{y}$:

$$\hat{y} = \mathrm{sigmoid}(\mathbf{W}_f \mathbf{h} + \mathbf{b}_f), \tag{32}$$

where $\mathbf{W}_f \in \mathbb{R}^{1\times d'}$ represents the trainable weight parameters and $\mathbf{b}_f$ represents the bias term. Finally, the model is trained using the cross-entropy function

$$\mathcal{L}(\Theta_m) = -\sum(y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)), \tag{33}$$

where Θ_m denotes the trainable parameters of the model, y_i represents the true label of event i, and $\hat{y}_i$ represents the corresponding predicted label.

5. Experiments

This section extensively evaluates the proposed model to demonstrate its effectiveness in the task of rumor detection. Specifically, Section 5.1 introduces the datasets and preprocessing methods. Section 5.2 presents the baseline models used for comparison. The experimental parameter settings are outlined in Section 5.3, while Section 5.4 showcases the experimental results and analysis.

5.1. Datasets and Preprocessing

There are two highly representative real-world and publicly available datasets constructed from Twitter and Weibo in the rumor detection task. Each dataset consists of a series of news events, with each event belonging to categories such as real news or fake news. Below is the introduction to the datasets:

- Weibo: Initially proposed in [7], data were captured from Sina Weibo, a popular online social media platform in China. This dataset contains comprehensive event information, including text, timestamps, and user configurations, all stored in the JSON file format. It consists a total of 2351 real news and 2312 fake news instances.
- Twitter 15 and Twitter 16: First introduced in [38], data were collected from the widely used online social networking platform Twitter. Each dataset consists of news events categorized into four classes: unverified, non-rumor, true, and false. These datasets include only the IDs of the tweets. We collected additional information such as reply texts and timestamps using the Twitter API.

We performed some simple preprocessing on the Weibo, Twitter 15, and Twitter 16 datasets. Subsequently, the preprocessed Twitter 15 and Twitter 16 datasets were merged into one dataset named Twitter, and the final datasets are described in Table 2.

For Twitter 15 and Twitter 16, we initially removed invalid tweets caused by user deletions or account suspensions. When constructing propagation graphs based on the reply relationships, we directly linked tweets with missing parent nodes to their source tweets. Next, we extracted events from Twitter 15 and Twitter 16 belonging to the non-rumors and true rumors categories, considering them true news and fake news, respectively.

These events were combined to create a larger dataset called Twitter, containing 562 fake news and 575 true news instances.

Table 2. Dataset details.

Statistic	Weibo	Twitter
Fake news number	2133	562
Real news number	2209	575
Time length	1576 h	159 h
Avg. number of tweets	378	25
Max. number of tweets	2000	256
Min. number of tweets	10	2

As for the Weibo dataset, due to limitations in computational resources, we had to remove events with over 2000 nodes. Similarly, we constructed the corresponding propagation graph based on the reply relationships. Finally, this resulted in a dataset comprising 2133 fake news and 2209 true news instances.

5.2. Baselines

The baseline models used for comparison with the proposed model in this paper are as follows, and Table 3 presents the feature types used by each model:

- SVM-RBF [5]: A method based on SVM with a radial basis function (RBF) kernel. It utilizes a range of statistical features from tweets to identify fake news.
- SVM-TS [6]: A linear SVM-based classifier that employs time series modeling techniques to capture the temporal features of rumors.
- GCN [59]: A graph representation learning method that uses message passing to aggregate information from neighboring nodes for feature extraction.
- GAT [49]: An advanced graph representation learning framework. Similar to a GCN, it incorporates attention mechanisms to differentiate the importance of different nodes.
- BU-RvNN [9]: A rumor classification method based on bottom-up recursive neural networks. It integrates text content and propagation structure features using GRUs and performs classification based on the state of the root node.
- TD-RvNN [9]: A rumor classification method based on top-down recursive neural networks. It integrates text content and propagation structure features using GRUs and performs classification based on the state of the leaf nodes.
- STS-NN [13]: A rumor detection method based on deep spatiotemporal neural networks. It integrates rumor propagation and temporal features within a GRU-like unit for learning.
- BiGCN [11]: A model based on GCNs that models rumor events separately using propagation and diffusion structures, followed by a Bidirectional GCN for feature extraction.

5.3. Experimental Set-up

In the experiment of this section, all SVM-based methods were implemented using sklearn, while all deep learning-based methods were implemented using PyTorch. The parameters were optimized using the Adam optimizer [60] with an initial learning rate of 5×10^{-5} and weight decay of 5×10^{-4}. Similar to a previous work [13], the entire dataset was divided into training, validation, and testing sets at an 8:1:1 ratio. The model with the best performance on the validation set was selected for testing. To ensure fair comparison, all models that utilized textual information employed the average of all word embeddings from the first layer encoder and the last layer encoder of the pretrained BERT model (first-to-last layer average), resulting in a tweet embedding of dimension 768. It was observed that the first-to-last layer average approach yielded significant performance improvements compared with using the CLS token or averaging only the last layer, particularly on the English datasets.

Table 3. Types of features used by different algorithms.

Method	Statistic	Text	Time	Structure
SVM-RBF	√			
SVM-TS	√		√	
GCN		√		√
GAN		√		√
TD-RvNN		√		√
BU-RvNN		√		√
STS-NN		√	√	√
BiGCN		√		√
SMGaDTP		√	√	√

In the SC-MGAT module, the parameter $\zeta(l)$ was set to $2l-1$, where $l \in [1,6]$. This means that each GNN layer utilized multi-hop neighbors with an interval of 1 (1-hop, 3-hop, 5-hop, ..., 11-hop), resulting in a total of six message-passing layers. For the DTP module, the window size ω was set to four, and the number of layers in LWA was set to two. For all datasets, the evaluation metrics used were precision (Prec.), recall (Rec.), F1 score (F1), and accuracy (Acc.) based on the predicted results. For each sample $c_i \in \mathcal{C}$, the differences between the predicted values and true values were measured using true positive (TP), false positive (FP), false negative (FN), and true negative (TN) results, and the metrics were calculated using the following formulas:

$$\text{Acc.} = \frac{TP+TN}{TP+FP+TN+FN}, \tag{34}$$

$$\text{Prec.} = \frac{TP}{TP+FP}, \tag{35}$$

$$\text{Rec.} = \frac{TP}{TP+FN}, \tag{36}$$

$$\text{F1} = \frac{2 \times \text{Prec.} \times \text{Rec.}}{\text{Prec.} + \text{Rec.}} \tag{37}$$

5.4. Experimental Results and Analysis

5.4.1. Comparison of Model Performance

Table 4 presents the evaluation results of all methods using the Weibo dataset and the Twitter dataset, where the best model is marked with bold formatting. From the table, the following conclusions can be drawn:

- Overall, our model outperformed other baseline methods in all datasets. On the Weibo dataset, our model improved the Acc and F1 by 1.39% and 1.42%, respectively, compared with the best baseline. On the Twitter dataset, the improvements were 3.51% and 3.57%, respectively. This confirms that our model effectively extracted more features compared with the baseline models, demonstrating the importance of high-order neighbor interaction features and differential temporal information in rumor detection tasks.
- The traditional machine learning-based methods exhibited lower performance across all datasets compared with the deep learning-based methods. This is because traditional methods rely on manually selected features, while deep learning algorithms can capture complex high-order features. Moreover, traditional machine learning only utilizes statistical-level features for text content, making it difficult to model semantic information.
- Consistent with [9] and others' findings, BU-RvNN performed worse than TD-RvNN. This is because BU-RvNN compresses features into a single node representation, resulting in significant information loss. In contrast, TD-RvNN performs pooling on

all leaf nodes to obtain the final features, thereby retaining more useful information. STS-NN utilizes both temporal and structural features simultaneously, but it still failed to achieve satisfactory results. This is partly due to compressing all the information into the last node.

- Among all the baseline models, BiGCN demonstrated stronger performance. Despite not utilizing temporal information, its ability to extract propagation and diffusion features enabled better structural information learning compared with STS-NN and TD-RvNN. However, the lack of high-order neighbor information (stacking only two layers of the GCN, capturing at most the interaction features of the two-hop neighbors) and the lack of temporal information restricted its performance.

Table 4. Rumor detection results of all algorithms.

Dataset	Method	Acc.	Prec.	Rec.	F1
Weibo	SVM-RBF	0.8134	0.7925	0.8195	0.8058
	SVM-TS	0.8295	0.7741	0.9024	0.8333
	GCN	0.9286	0.9223	0.9268	0.9246
	GAT	0.9470	0.9333	0.9561	0.9446
	TD-RvNN	0.9585	0.9431	0.9707	0.9567
	BU-RvNN	0.8963	0.9000	0.8780	0.8889
	STS-NN	0.9332	0.9314	0.9268	0.9291
	BiGCN	0.9447	0.9330	0.9512	0.9420
	SMGaDTP	**0.9724**	**0.9662**	**0.9756**	**0.9709**
Twitter	SVM-RBF	0.8142	0.8462	0.7719	0.8074
	SVM-TS	0.7727	0.7895	0.7759	0.7826
	GCN	0.8421	0.8136	0.8727	0.8421
	GAT	0.8860	0.8889	0.8727	0.8807
	TD-RvNN	0.8596	0.8305	0.8909	0.8596
	BU-RvNN	0.8509	0.8065	0.9091	0.8547
	STS-NN	0.8684	0.8571	0.8727	0.8649
	BiGCN	0.8947	0.8772	0.9091	0.8929
	SMGaDTP	**0.9298**	**0.9123**	**0.9455**	**0.9286**

5.4.2. Ablation Study

In this section, ablation experiments were conducted to explore the contributions of different modules in the model. Specifically, the following variants were proposed for comparison with the original model:

- SMGaDTP w/o SC-MGAT: This variant removes the SC-MGAT module and retains only the DTP module.
- SMGaDTP w/o DTP: This variant removes the DTP module and retains only the SC-MGAT module.
- SMGaDTP w/o DTP + SCA: This variant removes both the DTP and SCA modules, retaining only the Multi-hop GAT module.

Figure 5 displays the comparison results of different variants, and based on these results, the following conclusions can be drawn:

- Overall, SMGaDTP performed better than the other variants on all datasets, indicating that all the proposed modules play indispensable roles in rumor detection tasks.
- By comparing the w/o SC-MGAT, w/o DTP, and SMGaDTP variants, it can be observed that the w/o SC-MGAT variant performed the worst among all the variants, while the w/o DTP variant performed the best. This suggests that the structural features of rumors are more significant than temporal features, and the SC-MGAT module effectively captured the structural features of the rumors.
- By comparing the w/o DTP and w/o DTP + SCA variants, it can be inferred that the SCA module plays an important role in SC-MGAT, enhancing its ability to learn high-order neighbor information.

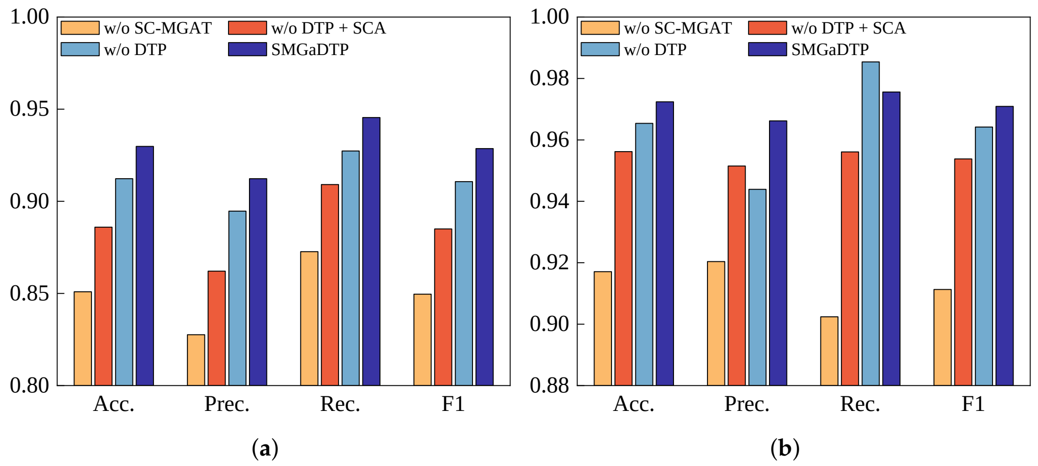

Figure 5. (**a**) Results of ablation experiments on Twitter dataset. (**b**) Results of ablation experiments on Weibo dataset.

5.4.3. Comparison of Early Detection

Early detection of rumors can significantly mitigate the damage caused by their spread. Numerous scholars have employed differential equations and numerical simulation methods to study the rate of rumor propagation and the control strategies [61–63]. These studies have all indicated that rumor dissemination exhibits a significant outbreak period, and early intervention in rumors can greatly reduce their scale and destructiveness. Additionally, some researchers have explored the sentiment scope on social media and pointed out that fake news generates intense negative emotions with considerable aggressiveness and stability. Furthermore, it tends to spread widely within the social space over time [64–66]. All of these studies have demonstrated the utmost necessity of intervening early in the case of rumors.

Therefore, SMGaDTP and all well-performing baseline models were subjected to early rumor detection capability testing. Specifically, early rumor detection testing requires setting a series of truncation times where only tweets published before the truncation time are used to test the detection capabilities of all models. In order to demonstrate the early detection capabilities of all models comprehensively, the truncation time on the Twitter dataset was set to {10 min, 20 min, 30 min, 40 min, 50 min, 1 h, 2 h, 3 h}, and the truncation time on the Weibo dataset was set to {2 h, 4 h, 6 h, 8 h, 10 h, 12 h, 24 h, 36 h}.

The results of the early detection are shown in Figure 6. The models that used both structural and temporal features are marked with solid lines, while the models that only used structural features are represented with dashed lines. It can be observed that on all datasets, as time progressed, our model generally outperformed the other baseline models. Additionally, it can be noted that the models using temporal features exhibited a stronger dependence on temporal information, with significant fluctuations in identification accuracy as time advanced. However, regardless of the model, both on the Twitter and Weibo datasets, optimal detection performance could be achieved after 1 h and 12 h, respectively. This indicates that all models could effectively identify early-stage rumors.

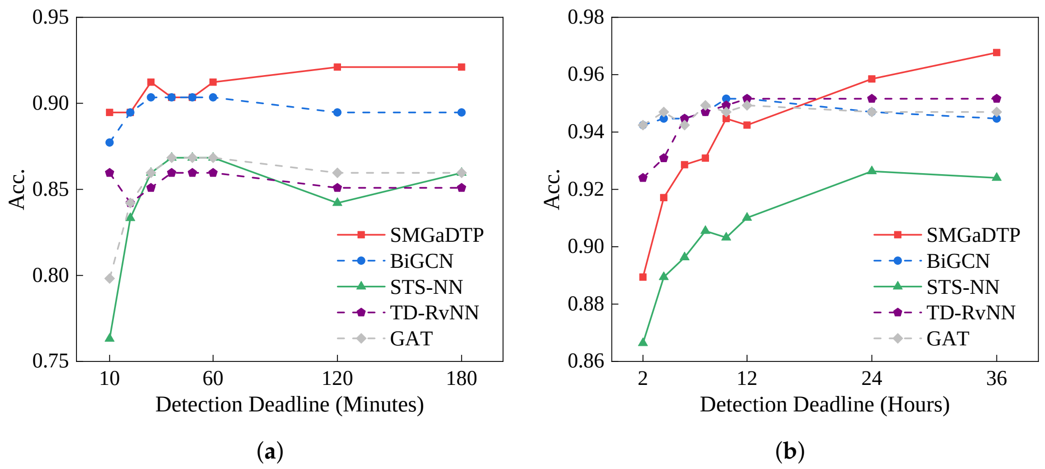

Figure 6. (**a**) Results of early detection on Twitter dataset. (**b**) Results of early detection on Weibo dataset.

5.4.4. Comparison of Deep Graph Detection

In this section, 818 events (409 positive and 409 negative samples) with relatively long node path relationships from the Weibo dataset were selected to demonstrate the differences between SC-MGAT and GAT in deep graphing. These events were used to train and test SC-MGAT and GAT, and all models used the same training parameters and experimental settings. The results are shown in Figure 7, where GAT represents a single-layer GAT and 6-GAT denotes stacking six layers of GAT. From the results, it can be observed that simply stacking multiple layers of GAT led to a decrease in performance compared with a single layer of GAT. On the other hand, SC-MGAT outperformed both the single-layer GAT and the stacked 6-GAT in all evaluation metrics. This further validates the effectiveness of our proposed approach.

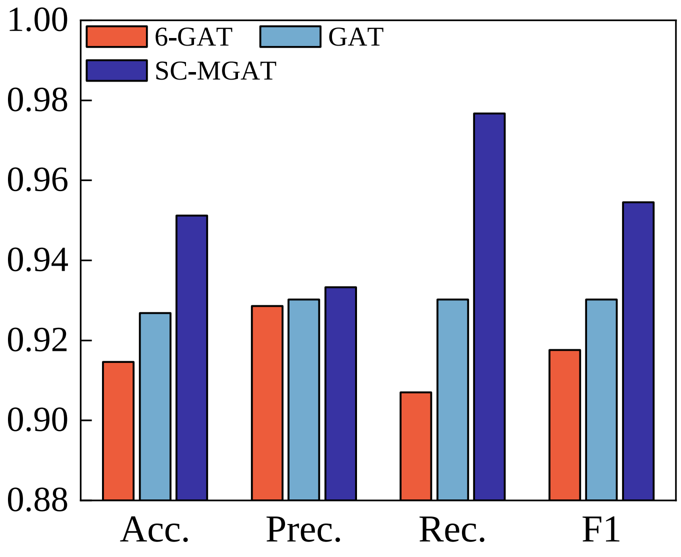

Figure 7. Comparison of SC-MGAT, GAT, and multilayer GAT.

In addition, we randomly selected a sample from the test set and applied multi-layer message passing using SC-MGAT and GAT. Afterward, for the nodes that had undergone message passing, the cosine similarity between each node and all other nodes was computed. The final similarity of each node in the graph was obtained by summing up its cosine similarities with all other nodes.

Figures 8 and 9 illustrate the visualization results, where nodes with higher similarity correspond to colors closer to deep red. From the visualization results, it can be observed that the traditional GAT exhibited oversmoothing phenomena after only three layers of message passing. This is consistent with the experimental results that showed a performance decrease when stacking multiple layers of GAT. In contrast, SC-MGAT maintained a higher level of node discrimination even after six layers of message passing, effectively avoiding the problem of oversmoothing.

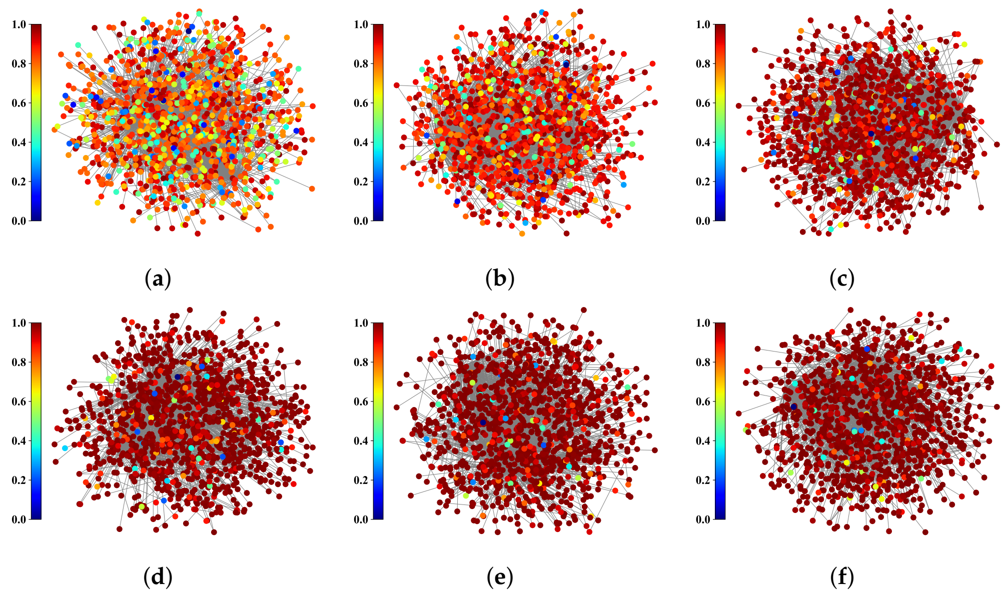

Figure 8. Node similarity after message passing with 6-GAT, where (**a**–**f**) represent the results from the first layer to the sixth layer, respectively.

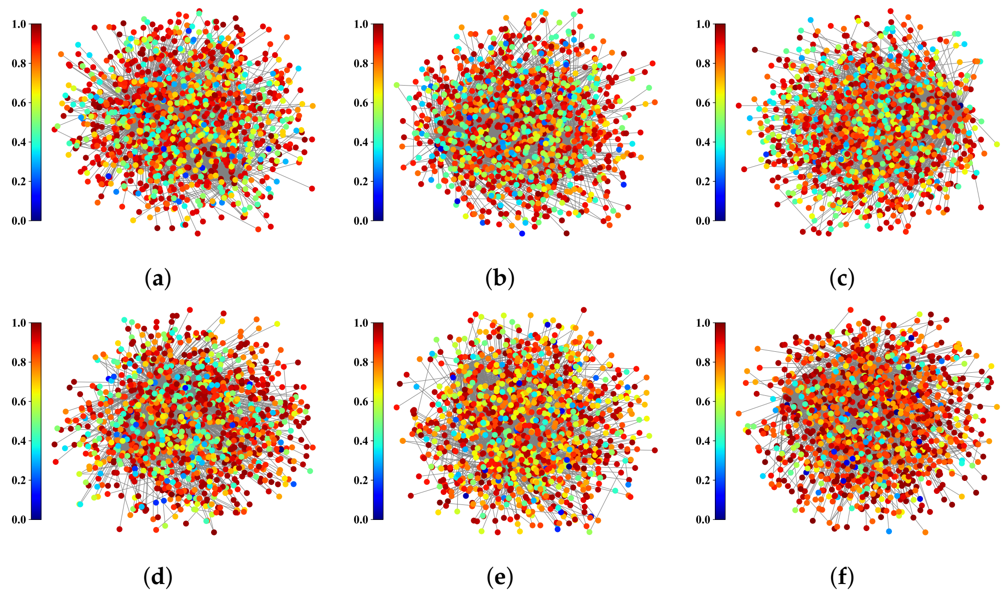

Figure 9. Node similarity after message passing with SC-MGAT, where (**a**–**f**) represent the results from the first layer to the sixth layer, respectively.

6. Conclusions

This paper proposes the SMGaDTP based on multi-hop graphing and differential time series, which consists of two parallel parts: SC-MGAT and DTP. SC-MGAT learns the structural features of rumors, while DTP learns the temporal features of rumors. We extensively tested the proposed model on widely used real-world Weibo and Twitter datasets, and the encouraging results demonstrated the effectiveness of our proposed model. This indicates that high-order structural information and differential temporal information can serve as effective features for rumor identification. Meanwhile, the ablation study demonstrates the respective contributions of each component of the model. Early detection indicates the model's capability to effectively recognize early-stage rumors. A comparison of SC-MGAT and GAT on the deep graph further confirmed their significant improvements in addressing the oversmoothing issue.

However, this study also has some limitations. First, the SC-MGAT module is built upon the homogeneous graph and has not been extended to the heterogeneous graph, limiting the model's ability to extract richer heterogeneous information. Secondly, it only models static graphs, making it challenging to capture the dynamic changes in event propagation structures over time. As for the DTP module, it focuses solely on temporal differences, neglecting the structural aspects. In the future, it can be expanded and integrated into the structural dimension to achieve a more comprehensive understanding. Lastly, concerning the fusion of temporal and structural features, this paper only employed a simple fully connected layer, which may have resulted in insufficient feature integration. Future works can explore more advanced methods, such as attention mechanisms and CNNs, to achieve a more comprehensive fusion of spatiotemporal features. These innovative approaches could potentially lead to improved performance and better understanding of the underlying dynamics in the data.

Author Contributions: Conceptualization, J.C.; methodology, J.C. and W.Z.; software, W.Z.; validation, J.C., W.Z. and S.Y.; formal analysis, W.Z. and H.M.; resources, J.C. and S.Y.; data curation, W.Z.; writing—original draft preparation, W.Z.; writing—review and editing, S.Y. and H.M.; visualization, W.Z.; supervision, S.Y.; project administration, S.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation Project of China under Grant No. 52274163.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data used to support the findings of this study are available from the corresponding author upon request.

Acknowledgments: The authors would like to express their thanks to the National Natural Science Foundation Project of China under Grant No. 52274163.

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

References

1. Sharma, K.; Qian, F.; Jiang, H.; Ruchansky, N.; Zhang, M.; Liu, Y. Combating Fake News: A Survey on Identification and Mitigation Techniques. *ACM Trans. Intell. Syst. Technol.* **2019**, *10*, 1–42. [CrossRef]
2. Guo, B.; Ding, Y.; Yao, L.; Liang, Y.; Yu, Z. The Future of False Information Detection on Social Media: New Perspectives and Trends. *ACM Comput. Surv.* **2020**, *53*, 1–36. [CrossRef]
3. Castillo, C.; Mendoza, M.; Poblete, B. Information Credibility on Twitter. In Proceedings of the 20th International Conference on World Wide Web, Hyderabad, India, 28 March–1 April 2011; pp. 675–684. [CrossRef]
4. Kwon, S.; Cha, M.; Jung, K.; Chen, W.; Wang, Y. Prominent Features of Rumor Propagation in Online Social Media. In Proceedings of the 2013 IEEE 13th International Conference on Data Mining, Dallas, TX, USA, 7–10 December 2013; pp. 1103–1108. [CrossRef]
5. Yang, F.; Liu, Y.; Yu, X.; Yang, M. Automatic Detection of Rumor on Sina Weibo. In Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics, Beijing, China, 12–16 August 2012; pp. 1–7. [CrossRef]
6. 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 on Conference on Information and Knowledge Management, Melbourne, Australia, 18–23 October 2015; pp. 1751–1754. [CrossRef]
7. 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 Twenty-Fifth International Joint Conference on Artificial Intelligence, New York, NY, USA, 9–15 July 2016; pp. 3818–3824.
8. Yu, F.; Liu, Q.; Wu, S.; Wang, L.; Tan, T. A Convolutional Approach for Misinformation Identification. In Proceedings of the 26th International Joint Conference on Artificial Intelligence, Melbourne, VIC, Australia, 19–25 August 2017; pp. 3901–3907. [CrossRef]

9. Ma, J.; Gao, W.; Wong, K.F. Rumor Detection on Twitter with Tree-structured Recursive Neural Networks. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Melbourne, VIC, Australia, 15–20 July 2018; pp. 1980–1989. [CrossRef]
10. Monti, F.; Frasca, F.; Eynard, D.; Mannion, D.; Bronstein, M.M. Fake news detection on social media using geometric deep learning. *arXiv* **2019**, arXiv:1902.06673.
11. Bian, T.; Xiao, X.; Xu, T.; Zhao, P.; Huang, W.; Rong, Y.; Huang, J. Rumor Detection on Social Media with Bi-Directional Graph Convolutional Networks. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; pp. 549–556. [CrossRef]
12. Lin, H.; Zhang, X.; Fu, X. A Graph Convolutional Encoder and Decoder Model for Rumor Detection. In Proceedings of the 2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA), Sydney, NSW, Australia, 6–9 October 2020; pp. 300–306. [CrossRef]
13. Huang, Q.; Zhou, C.; Wu, J.; Liu, L.; Wang, B. Deep spatial–temporal structure learning for rumor detection on Twitter. *Neural Comput. Appl.* **2020**, *35*, 12995–13005. [CrossRef]
14. Zhao, L.; Akoglu, L. Pairnorm: Tackling oversmoothing in gnns. *arXiv* **2019**, arXiv:1909.12223.
15. Li, G.; Muller, M.; Thabet, A.; Ghanem, B. Deepgcns: Can gcns go as deep as cnns? In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 29 October–2 November 2019; pp. 9267–9276. [CrossRef]
16. Choi, J.; Ko, T.; Choi, Y.; Byun, H.; Kim, C.k. Dynamic graph convolutional networks with attention mechanism for rumor detection on social media. *PLoS ONE* **2021**, *16*, e0256039. [CrossRef]
17. Song, C.; Shu, K.; Wu, B. Temporally evolving graph neural network for fake news detection. *Inf. Process. Manag.* **2021**, *58*, 102712. [CrossRef]
18. Song, C.; Teng, Y.; Zhu, Y.; Wei, S.; Wu, B. Dynamic graph neural network for fake news detection. *Neurocomputing* **2022**, *505*, 362–374. [CrossRef]
19. Sun, M.; Zhang, X.; Zheng, J.; Ma, G. Ddgcn: Dual dynamic graph convolutional networks for rumor detection on social media. In Proceedings of the AAAI Conference on Artificial Intelligence, Online, 22 February–1 March 2022; pp. 4611–4619. [CrossRef]
20. Chen, T.; Li, X.; Yin, H.; Zhang, J. Call attention to rumors: Deep attention based recurrent neural networks for early rumor detection. In Proceedings of the Trends and Applications in Knowledge Discovery and Data Mining: PAKDD 2018 Workshops, Melbourne, VIC, Australia, 3 June 2018; pp. 40–52. [CrossRef]
21. Wang, Z.; Guo, Y.; Li, Z.; Tang, M.; Qi, T.; Wang, J. Research on microblog rumor events detection via dynamic time series based GRU model. In Proceedings of the ICC 2019-2019 IEEE International Conference on Communications (ICC), Shanghai, China, 20–24 May 2019; pp. 1–6. [CrossRef]
22. Fournier, L.; Dupoux, E.; Dunbar, E. Analogies minus analogy test: Measuring regularities in word embeddings. *arXiv* **2020**, arXiv:2010.03446.
23. Puccetti, G.; Miaschi, A.; Dell'Orletta, F. How Do BERT Embeddings Organize Linguistic Knowledge? In Proceedings of the Deep Learning Inside out (DeeLIO): The 2nd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures, Online, 20–24 June 2021; pp. 48–57. [CrossRef]
24. Dalvi, F.; Durrani, N.; Sajjad, H.; Belinkov, Y.; Bau, A.; Glass, J. What Is One Grain of Sand in the Desert? Analyzing Individual Neurons in Deep NLP Models. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; pp. 6309–6317. [CrossRef]
25. Durrani, N.; Sajjad, H.; Dalvi, F.; Belinkov, Y. Analyzing individual neurons in pre-trained language models. *arXiv* **2020**, arXiv:2010.02695.
26. Rong, Y.; Huang, W.; Xu, T.; Huang, J. The truly deep graph convolutional networks for node classification. *arXiv* **2019**, arXiv:1907.10903.
27. Takahashi, T.; Igata, N. Rumor detection on twitter. In Proceedings of the 6th International Conference on Soft Computing and Intelligent Systems, and the 13th International Symposium on Advanced Intelligence Systems, Kobe, Japan, 20–24 November 2012; pp. 452–457. [CrossRef]
28. Ratkiewicz, J.; Conover, M.; Meiss, M.; Gonçalves, B.; Patil, S.; Flammini, A.; Menczer, F. Detecting and tracking the spread of astroturf memes in microblog streams. *arXiv* **2010**, arXiv:1011.3768.
29. Yang, Y.; Zheng, L.; Zhang, J.; Cui, Q.; Li, Z.; Yu, P.S. TI-CNN: Convolutional neural networks for fake news detection. *arXiv* **2018**, arXiv:1806.00749.
30. Shu, K.; Cui, L.; Wang, S.; Lee, D.; Liu, H. defend: Explainable fake news detection. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, AK, USA, 4–8 August 2019; pp. 395–405. [CrossRef]
31. Khoo, L.M.S.; Chieu, H.L.; Qian, Z.; Jiang, J. Interpretable Rumor Detection in Microblogs by Attending to User Interactions. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; pp. 8783–8790. [CrossRef]
32. Hosseinimotlagh, S.; Papalexakis, E.E. Unsupervised content-based identification of fake news articles with tensor decomposition ensembles. In Proceedings of the Workshop on Misinformation and Misbehavior Mining on the Web (MIS2), Los Angeles, CA, USA, 9 February 2018.

33. Wang, Y.; Ma, F.; Jin, Z.; Yuan, Y.; Xun, G.; Jha, K.; Su, L.; Gao, J. Eann: Event adversarial neural networks for multi-modal fake news detection. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, UK, 19–23 August 2018; pp. 849–857. [CrossRef]
34. Gautam, A.; Venktesh, V.; Masud, S. Fake news detection system using xlnet model with topic distributions: Constraint@ aaai2021 shared task. In Proceedings of the CONSTRAINT Shared Task in AAAI-2021, Online, 8 February 2018; pp. 189–200. [CrossRef]
35. Ghanem, B.; Ponzetto, S.P.; Rosso, P.; Rangel, F. Fakeflow: Fake news detection by modeling the flow of affective information. *arXiv* **2021**, arXiv:2101.09810.
36. Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J.; Salakhutdinov, R.R.; Le, Q.V. Xlnet: Generalized autoregressive pretraining for language understanding. In Proceedings of the 33rd International Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; pp. 5753–5763.
37. Zhang, X.; Cao, J.; Li, X.; Sheng, Q.; Zhong, L.; Shu, K. Mining Dual Emotion for Fake News Detection. In Proceedings of the Web Conference 2021, Ljubljana, Slovenia, 19–23 April 2021; pp. 3465–3476. [CrossRef]
38. Ma, J.; Gao, W.; Wong, K.F. Detect rumors in microblog posts using propagation structure via kernel learning. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Vancouver, BC, Canada, 30 July–4 August 2017; pp. 708–717. [CrossRef]
39. Benamira, A.; Devillers, B.; Lesot, E.; Ray, A.K.; Saadi, M.; Malliaros, F.D. Semi-Supervised Learning and Graph Neural Networks for Fake News Detection. In Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, Vancouver, BC, Canada, 27–30 August 2019; pp. 568–569. [CrossRef]
40. Yuan, C.; Ma, Q.; Zhou, W.; Han, J.; Hu, S. Jointly embedding the local and global relations of heterogeneous graph for rumor detection. In Proceedings of the 2019 IEEE International Conference on Data Mining (ICDM), Beijing, China, 8–11 November 2019; pp. 796–805. [CrossRef]
41. Huang, Q.; Yu, J.; Wu, J.; Wang, B. Heterogeneous Graph Attention Networks for Early Detection of Rumors on Twitter. In Proceedings of the 2020 International Joint Conference on Neural Networks (IJCNN), Glasgow, UK, 19–24 July 2020; pp. 1–8. [CrossRef]
42. Tu, K.; Chen, C.; Hou, C.; Yuan, J.; Li, J.; Yuan, X. Rumor2vec: A rumor detection framework with joint text and propagation structure representation learning. *Inf. Sci.* **2021**, *560*, 137–151. [CrossRef]
43. Dong, M.; Zheng, B.; Quoc Viet Hung, N.; Su, H.; Li, G. Multiple Rumor Source Detection with Graph Convolutional Networks. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management, Beijing, China, 3–7 November 2019; pp. 569–578. [CrossRef]
44. Xu, K.; Hu, W.; Leskovec, J.; Jegelka, S. How powerful are graph neural networks? *arXiv* **2018**, arXiv:1810.00826.
45. Chen, Y.; You, J.; He, J.; Lin, Y.; Peng, Y.; Wu, C.; Zhu, Y. SP-GNN: Learning structure and position information from graphs. *Neural Netw.* **2023**, *161*, 505–514. [CrossRef]
46. You, J.; Gomes-Selman, J.M.; Ying, R.; Leskovec, J. Identity-aware Graph Neural Networks. In Proceedings of the AAAI Conference on Artificial Intelligence, Online, 2–9 February 2021; pp. 10737–10745. [CrossRef]
47. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778. [CrossRef]
48. Xue, H.; Sun, X.K.; Sun, W.X. Multi-hop hierarchical graph neural networks. In Proceedings of the 2020 IEEE International Conference on Big Data and Smart Computing (BigComp), Busan, Republic of Korea, 19–22 February 2020; pp. 82–89. [CrossRef]
49. Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; Bengio, Y. Graph attention networks. *arXiv* **2017**, arXiv:1710.10903.
50. Brody, S.; Alon, U.; Yahav, E. How attentive are graph attention networks? *arXiv* **2021**, arXiv:2105.14491.
51. Knyazev, B.; Taylor, G.W.; Amer, M. Understanding Attention and Generalization in Graph Neural Networks. In Proceedings of the 33rd International Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; pp. 4202–4212.
52. Ying, R.; You, J.; Morris, C.; Ren, X.; Hamilton, W.L.; Leskovec, J. Hierarchical Graph Representation Learning with Differentiable Pooling. In Proceedings of the 32nd International Conference on Neural Information Processing Systems, Montréal, QC, Canada, 3–8 December 2018; pp. 4805–4815.
53. Ranjan, E.; Sanyal, S.; Talukdar, P. ASAP: Adaptive Structure Aware Pooling for Learning Hierarchical Graph Representations. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; pp. 5470–5477. [CrossRef]
54. Fan, X.; Gong, M.; Wu, Y.; Qin, A.K.; Xie, Y. Propagation Enhanced Neural Message Passing for Graph Representation Learning. *IEEE Trans. Knowl. Data Eng.* **2023**, *35*, 1952–1964. [CrossRef]
55. Xu, K.; Li, C.; Tian, Y.; Sonobe, T.; Kawarabayashi, K.i.; Jegelka, S. Representation Learning on Graphs with Jumping Knowledge Networks. In Proceedings of the 35th International Conference on Machine Learning, Stockholmsmässan, Stockholm, Sweden, 10–15 July 2018; pp. 5453–5462.
56. Karim, F.; Majumdar, S.; Darabi, H.; Harford, S. Multivariate LSTM-FCNs for time series classification. *Neural Netw.* **2019**, *116*, 237–245. [CrossRef] [PubMed]
57. Chen, R.; Yan, X.; Wang, S.; Xiao, G. DA-Net: Dual-attention network for multivariate time series classification. *Inf. Sci.* **2022**, *610*, 472–487. [CrossRef]

58. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is All You Need. In Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 6000–6010.
59. Kipf, T.N.; Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv* **2016**, arXiv:1609.02907.
60. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
61. Chen, G. ILSCR rumor spreading model to discuss the control of rumor spreading in emergency. *Phys. A* **2019**, *522*, 88–97. [CrossRef]
62. Cheng, Y.; Huo, L.; Zhao, L. Dynamical behaviors and control measures of rumor-spreading model in consideration of the infected media and time delay. *Inf. Sci.* **2021**, *564*, 237–253. [CrossRef]
63. Li, D.; Hu, P.; Guan, Z.H.; Li, T. An Efficient Hybrid Control Strategy for Restraining Rumor Spreading. *IEEE Trans. Syst. Man Cybern. Syst.* **2021**, *51*, 6779–6791. [CrossRef]
64. Bonifazi, G.; Cauteruccio, F.; Corradini, E.; Marchetti, M.; Sciarretta, L.; Ursino, D.; Virgili, L. A Space-Time Framework for Sentiment Scope Analysis in Social Media. *Big Data Cogn. Comput.* **2022**, *6*, 130. [CrossRef]
65. Schöne, J.P.; Parkinson, B.; Goldenberg, A. Negativity spreads more than positivity on Twitter after both positive and negative political situations. *Affect. Sci.* **2021**, *2*, 379–390. [CrossRef]
66. Yu, H.; Yang, C.C.; Yu, P.; Liu, K. Emotion diffusion effect: Negative sentiment COVID-19 tweets of public organizations attract more responses from followers. *PLoS ONE* **2022**, *17*, e0264794. [CrossRef]