DGCN-TES: Dynamic GCN-Based Multitask Model With Temporal Event Sharing for Rumor Detection

Shuzhen Wan, Guanghao Yang, Fangmin Dong, and Mengyuan Wang

***Abstract*—The rumor detection task aims to identify unofficial and unconfirmed information that is spreading on social media. At any given moment, different users express their opinions, focusing on some propagation events, and the posts they make gradually form a social network that expands as it grows. Over time, nodes and edges form a dynamic graph that presents different states at different moments. However, most existing research focuses more on the text content, social context, propagation mode, etc., and they ignore the factors from many aspects and do not consider the dynamic relationships implied in the propagation development of social media. To analyze these dynamic properties, this article proposes a dynamic network-based multitask rumor detection method called dynamic GCN-based multitask model with temporal event sharing for rumor detection (DGCN-TES). This method can effectively capture the dynamic patterns of relationships in propagation events and change them over time to detect rumors. It is mainly divided into three modules: 1) dynamic-graph convolutional network (GCN) module, which uses dynamic graph neural network to construct the propagation graph of rumor events at different times, which can better capture the dynamic spatial features that change over time; 2) content-long short-term memory (LSTM), which uses the LSTM network as a benchmark model and has been improved to better capture time-series text features over time and for multitask shared interactions; and 3) temporal event sharing layer is the sharing layer, which uses time step as the basic unit of sharing, and realizes the sharing interaction between dynamic structural features and temporal text features between the first two modules. We tested the method on two real-world rumor detection datasets PHEME and WEIBO, and the final results show that the method improved F1-score by more than 2.63% and 3.91% compared to the other best baselines baseline.**

***Index Terms*—Dynamic graph neural network, multitask, temporal event sharing, time step.**

I. Introduction

IN the field of social psychology, rumor refers to the circulation of information that has not been officially confirmed or news that is deliberately fabricated by human beings [1]. Recently, the Internet and social media platforms have been the main channels through which people obtain news, but the lack of censorship brings many opportunities for rumors to spread. For example, news about COVID-19[1] [2] is spreading rapidly (e.g., some self-media claim that drinking bleach can cure this disease) and is recognized as a rumor by the World Health Organization.[2] This kind of social opinion is very dangerous, and some rumors can cause social panic and serious consequences if they are not identified in time. Therefore, correctly distinguishing social media rumor events, maintaining public order of social opinion, strengthening supervision, and effectively screening rumor events are urgent tasks for today's social networks. With the development of artificial intelligence technology, deep learning has been widely applied to the field of rumor detection. Recent research [3], [4], [5] shows that by modeling the mode of spread of rumor events as a spreading tree or spread diagram, similarities in structure can be used to distinguish rumor and nonrumor. For example, by modeling news samples into graphs and sentence classification problems into graph classification problems [6], these methods demonstrate the point that static spatial structures can be used for rumor detection. The reason that this idea can detect rumors well is its ability to learn rumor event propagation patterns, which is not possible with methods related to textual content based [7]. However, this method of constructing communication patterns is static, and it can only reflect the communication patterns of rumor events at the late stage of development or when they have already ended, whereas the evolution of rumor events in the real world develops dynamically over time, and this method fails to capture the dynamic characteristics of rumor events in their constant changes over time.

Dynamic graph neural networks can be a good solution to the above problems. Some detection methods based on dynamic ideas have been proposed [8], [9], [10], which have better performance by simulating the propagation process of news events in the real world and capturing more delicate features that change over time. A common approach [10] tends to be to construct static graphs of the same propagation event, at different time stages, to form a set of static graph sequences that are continuous in time. The performance of this approach is limited by the number of time steps, and if the number of time steps is set too small, a better dynamic effect cannot be realized. Therefore, this article tries to construct a fine-grained dynamic

Manuscript received 27 August 2023; revised 16 April 2024 and 12 July 2024; accepted 31 July 2024. Date of publication 5 September 2024; date of current version 3 December 2024. This work was supported in part by the National Nature Science Foundation of China under Grant U1703261 and Grant 61871258. *(Corresponding author: Shuzhen Wan.)*

The authors are with Hubei Key Laboratory of Intelligent Vision-Based Monitoring for Hydroelectric Engineering, Yichang Key Laboratory of Intelligent Medicine, and the College of Computer and Information Technology, China Three Gorges University, Yichang 443002, Hubei, China (e-mail: wanshuzhen@163.com; guanghao_y@163.com; fmdong@ctgu.edu.cn; Meng_yuan_Wang@163.com).

Digital Object Identifier 10.1109/TCSS.2024.3443275

[1] https://thebulletin.org/2020/01/fake-news-epidemic-coronavirus-breeds-hate-and-disinformation-in-india-and-beyond/

[2] https://m.facebook.com/WHO/photos/fact-drinking-methanol-ethanol-or-bleach-does-not-prevent-or-cure-covid-19-and-c/3040991822612847/

graph neural network based on time steps, whose number of time steps corresponds to the number of postpropagation of rumor events, to capture a set of transition-dedicated time-based dynamic structural features, and such dynamic structural features have a certain relationship with the time sequence of recurrent neural networks in the time dimension.

Recurrent neural networks, as a common approach to solving rumor detection, are good at handling sequential data, taking into account the previous inputs when calculating each time step, and can capture and utilize the contextual information in sequential data well. Therefore, it has been widely utilized in recent years. For example, Ma et al. [11] used the recurrent neural network (RNN)-based approach to capture contextual information of related posts in microblog events over time. Ahmad et al. [12] constructed bidirectional long short-term memory (LSTM)-RNN networks for rumor prediction by learning social and content-based features. All of these methods treat textual content as sequence data, and there are contextual and temporal relationships between different texts of a sequence, a feature similar to the propagation process of a dynamic graph network, i.e., neighboring contextual tweets correspond to nodes where the dynamic graph is being propagated in neighboring time steps. If it were possible to achieve a fine-grained dynamic graph network based on time steps, then there would be a certain relationship between the sequence of diffusion of preceding and subsequent nodes in the dynamic graph, and the time sequence order between the upward and downward propagation of texts in the LSTM network. They would be able to capture the dynamic structural features of time steps and the temporal sequential text features, respectively. The above-mentioned detection approaches on sequential data rely only on a single feature such as content, but current social networks are multifaceted and changeable, and they do not take into account features such as relationships, behaviors, and communication structures in rumor propagation events.

From the above processing of dynamic graphs as well as LSTM, we are inspired by the fact that it is possible to combine the advantages of both from a dynamic point of view and cotrain them to learn a representation that combines dynamic structural features and temporal textual features. Meanwhile, the multitask learning approach can combine the features of multiple tasks well, and related works [13], [14], [15], [16] often define the LSTM layer as a shared network to combine other networks, they all use hardware parameter sharing to interact with features among other tasks, and although this approach achieves parameter sharing to some extent and can improve model performance, rich feature sharing among multiple tasks is neglected. Therefore, we choose the LSTM network as a shared layer to combine dynamic graph neural network and text-based LSTM network through multitasking, and the advantage is that we take the time step as the basic computational unit, and during the training, multiple dynamic modules start exchanging and learning each other's features at the same time step, to better improve prediction results.

In this article, we simulate the propagation process of news events in the real world to capture its dynamic features in a more suitable way, and we design a better sharing way to train the model to eventually distinguish rumors and nonrumors. The dynamic graph convolutional network (GCN)-based multitask model with temporal event sharing for rumor detection is proposed. It constructs rumor event propagation graphs corresponding to various temporal stages and captures the evolving dynamic structural features of rumor events by dynamic graph neural networks. Simultaneously, the textual content features are captured by LSTM networks. To integrate the two networks, we designed a novel sharing layer based on temporal steps. Finally, an attention mechanism is employed to update output features for rumor detection.

The contributions of this article are as follows.

1) A dynamic network-based multitask detection model (DGCN-TES) is proposed, which integrates dynamic graph neural network and LSTM in a multitask method, and learns information that combines dynamic structural features and temporal textual features, which effectively improves the performance of the model.
2) We propose a fine-grained dynamic graph neural network based on time steps, to capture dynamic structural features of rumor events with more delicate transitions to increase the efficiency of multitask sharing and ultimately improve the detection results.
3) A temporal event-based sharing method (TES) is proposed for multitasking interaction. Compared with the previous interaction methods, TES exerts better sharing performance, which interacts with the information of multitasks at each time step and effectively improves the prediction.
4) We conducted a series of experiments on two real-world datasets, and the experimental results demonstrate that our proposed dynamic GCN-based multitask model with temporal event sharing for rumor detection (DGCN-TES) is effective and outperforms other approaches.

II. Related Work

In this section, the work related to the proposed model is reviewed, and this research focuses on the following topics: text content-based detection methods, propagation structure-based detection methods, and multitask-based detection methods.

A. Text Content-Based Detection Methods

In the event of rumor spreading, it is often dependent on specific forms of social media, such as text, images, and video screens. These media-specific forms mainly include textual features [7] and visual features. This literature mainly discusses the description of textual content from the linguistic level, such as sentences, vocabulary, and semantics. After these contents are cleaned, deduplicated, and other operations, the potential text features can be obtained using text embedding techniques to represent sentences as vectors, which are used as inputs to neural networks. For example, Singh et al. [16] proposed an attention-based LSTM network that uses LSTM to model textual content and subsequently uses the text with several different language and user features to distinguish rumors. Ma et al. [11]

used data from recurrent neural time series for modeling, using an RNN network that can learn the feature of the sequential data to capture the features of rumor events over time. Ma et al. [17] proposed a generative adversarial network (GAN)-based model to obtain a feature representation of fake news, which is based on a GRU-based generator to generate controversial instances, and in turn, designed an RNN-based discriminator to identify instances. Cheng et al. [18] mainly proposed a GAN-based hierarchical framework for text-level rumor detection and provided solutions for interpretation and gene classification. However, most content-based detection methods rely only on a single feature, or content, and do not take into account the relationships and behaviors in a rumor-spreading event. Therefore, this article considers the integration of text-related work with other work in the form of multitasking to learn multifaceted features for prediction.

B. Propagation Structure-Based Detection Methods

In a large number of deep learning tasks, rumor propagation events often imply temporal and structural features. The process of rumor events is not static; it is a historical collection of information that is constantly changing over time, and the structure of propagation is also constantly changing. There is a slight difference between the propagation pattern of true news and the propagation process of false news: for example, true news tends to propagate more slowly [19], and the content of true news is richer and usually related but not identical content can appear in multiple posts; while the propagation speed of false news tends to be exploded growth in a short time, and the propagation content tends to be a few limited and roughly the same words and pictures, with a more scanty degree of content. The richness of the content is relatively scarce. Based on the above features, rumors can be analyzed from the perspectives of time and dissemination structure and then distinguished. Some approaches have explored studies based on the structural features (e.g., propagation features) of social networks [20]. Zhang et al. [21] proposed a lightweight propagation path aggregation neural network, where they modeled the propagation structure of each rumor as a set of independent propagation paths for rumor embedding and classification. For early detection of rumor events, Silva et al. [22] proposed training an autoencoder to learn the embedding of the entire propagation network, which they demonstrated would give better results for early rumor detection. Vaibhav et al. [6] modeled each article in the dataset as a graph and formulated the fake news detection task as a graph classification task, where the nodes of the graph come to represent the sentences of the article and the edges between the nodes represent the semantic similarity between the sentences. Zhang et al. [23] constructed a propagation graph by tracking the propagation structure of posts and proposed an algorithm based on gated graph neural networks to generate a robust representation for each node for rumor detection. Yang et al. [24] explored the rumor problem from an adversarial perspective based on graphs to extract more unique structural features for better rumor detection by dynamically generating perturbations on heterogeneous social graphs with domain constraints. Bian et al. [3] proposed a novel bidirectional graphical model (Bi-GCN) that explores these two features by running on top–down and bottom–up propagation of rumors.

The graph neural network effectively captures the global structural features learned during the propagation of a rumor event; however, the propagation graph of many applications changes over time during the development of real social media. Traditional graph neural networks have limited attention to temporal features, and numerous studies on dynamic graph neural networks have emerged to be able to capture both temporal and spatial structural features. Instead of learning with a static network, Song et al. [8] proposed a dynamic graph-based detection framework to simulate the event evolution of real-world news. Sun et al. [10] investigated a dynamic propagation graph-based news detection problem, where they utilized a structure-aware module and an event-aware module to capture temporal and network structure information, respectively. Huang et al. [25], although they did not construct a graph structure to deal with the problem, proposed a spatiotemporal structured neural network, which considers the spatial and temporal structures as a whole, to model the news propagation for rumor detection, taking full advantage of the temporal features that cannot be captured by traditional networks. A dual dynamic graph convolutional network was investigated in the literature [10], modeling spatial, temporal, and textual features in a single architecture, passing multiple fused messages to subsequent network units in a sequential manner.

Numerous studies have shown that dynamic graph neural networks possess more powerful capabilities compared to static graph networks, which not only capture the relational features of the propagated structure but also the dynamic structural features of the temporal sequence by constructing static graphs in the form of different temporal sequences.

Therefore, in this article, we choose a dynamic graph neural network as one of the baseline models. We try to construct a time-step-based fine-grained dynamic graph neural network.

C. Multitask-Based Detection Methods

Multitask learning refers to joint learning behaviors that can share information about multiple related but not identical tasks [26]. Auxiliary tasks can support the primary task in learning certain features that are easily overlooked or difficult to learn by the primary task itself, which makes multitask learning particularly important when certain potential, but more valuable, features are not well utilized by the primary task. Collobert and Weston [27] described a single convolutional neural network architecture that, given a sentence outputting multiple different predictions, shares the parameters of multitasks across the network for joint training. Kochkina et al. [15] constructed a multitask learning architecture consisting of three tasks dealing with truthfulness classification, stance classification, and rumor detection tasks, and in a concrete implementation, only one set of hidden layers was defined, and the same hidden layers were used for different tasks, allowing them to achieve hard

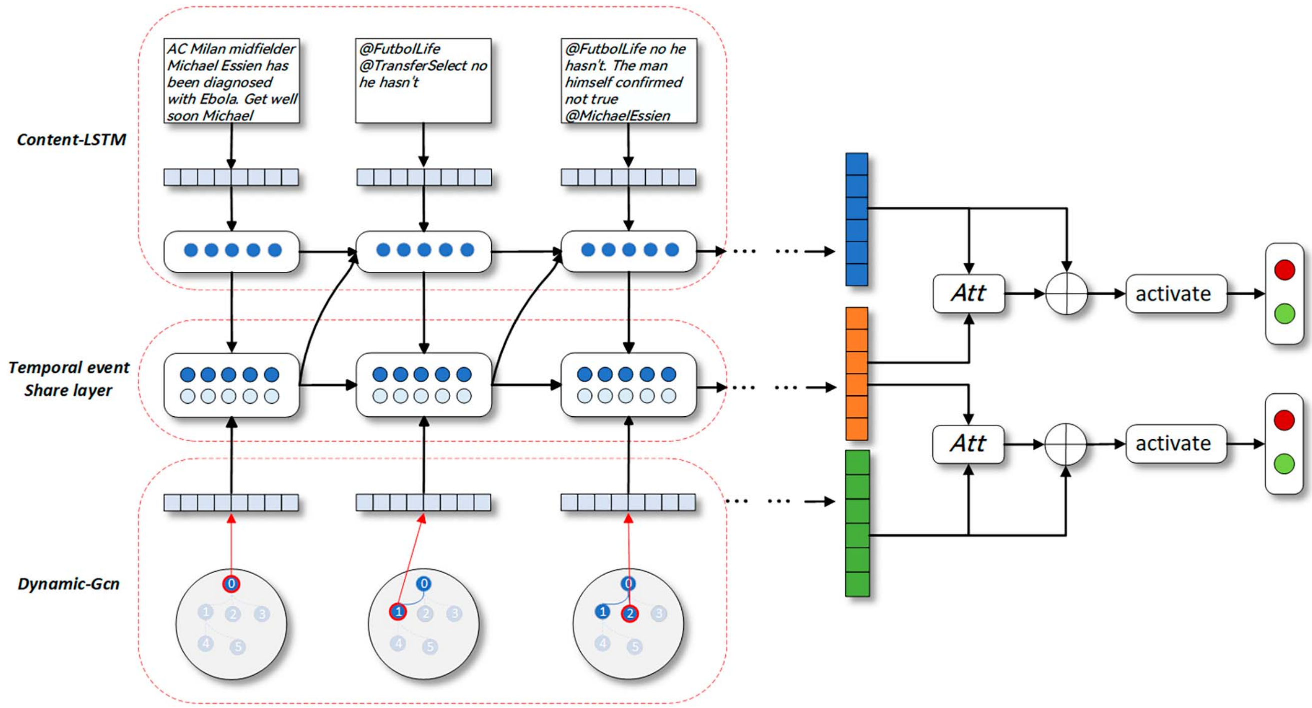

Fig. 1. Dynamic network-based multitask detection framework.

parameter sharing. Wan et al. [14] jointly performed rumor detection and stance classification by using two RNN-based network architectures built with shared layers. Wu et al. [28] explored shared layers with gating and attention mechanisms that can selectively capture valuable shared features for fake news detection and stance classification. Bai et al. [29] proposed a multitask attention tree neural network (MATNN) specifically designed to provide a structured representation of rumor conversations and exploited the attention mechanism to jointly classify stances and detect rumor veracity. Cheng et al. [30] proposed a multitask variational autoencoder-assisted rumor classification system consisting of four components: rumor detection, rumor tracking, stance classification, and veracity classification, where a more appropriate classification technique and training engine improves the performance and generalization of the model.

However, most of the existing multitask learning methods realize sharing only by defining a set of RNNs as a shared layer and using hard parameter sharing to interact with other intertask features, which, although it realizes parameter sharing to a certain extent and can improve the model performance, the feature sharing among multitasks is neglected.

Therefore, in this article, based on the multitask sharing layer, we constructed a shared network that can share multitask features as well as hard parameters simultaneously at the time of training.

III. DGCN-TES Model

In this article, we propose a dynamic network-based multitask detection model (DGCN-TES), which performs multitasking interaction for rumor detection in a dynamic perspective. It is mainly divided into four modules: dynamic-GCN, content-LSTM, temporal event sharing layer (hereinafter referred to as TES), and fusion and classification module, as shown in Fig. 1. First, the propagation graphs of rumor events at different moments are constructed, and they are modeled by using the dynamic graph convolutional network to capture the rumor events' dynamic structural features; second, the temporal share layer is used as a shared network combining dynamic-GCN and content-LSTM to interact with features at each identical time step; and finally, the attention mechanism is used to focus on the information that has an important impact on different tasks and between different moments, which is used as the input to the classifier to make predictions.

Compared to other related work [8], [10] on dynamic graphs, although they also consider dynamic graph neural networks to deal with the dynamic correlation patterns of rumor propagation events, the former does not yet adequately consider temporal granularity, and the latter neglects the dynamic interaction of multitasks. In contrast, the computational process of DGCN-TES proposed in this article follows a temporal event-sharing strategy based on time-step fine granularity, and this design creates a different snapshot of the dynamic graph at each moment, which not only improves the efficiency of multitask sharing but also makes the dynamic graphs smoother in terms of the temporal dimensions of the excess.

A. Problems and Symbolic Definitions

The rumor detection task is defined as a binary classification problem. Let $\varepsilon = \varepsilon_1, \ldots, \varepsilon_n$ be a set of instances of rumor detection events, ε_i is the ith event, and n is the number of propagation events to be detected in the dataset. Denote the set of posts

TABLE I
IMPORTANT NOTATIONS AND DESCRIPTION

Notations	Descriptions
ε_i	The *i*th event
ε_i^c	The set of tweets in the *i*th event
γ	The time step of different tweets in the *i*th event
V_{ir}, E_{ir}	Nodes and edges in the ith event at the moment γ
$\mathcal{G}_{ir} =< V_{ir}, E_{ir} >$	Propagation graph for the γ moment in the *i*th event

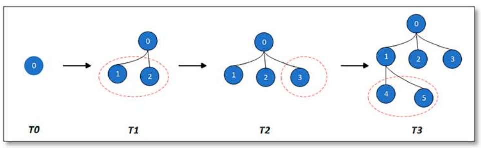

Fig. 2. Coarse-grained dynamic graphs based on time steps.

and comment contents as $\varepsilon_i^c = \{s_i, c_{i1}, ..., c_{im_{i-1}}\}$, s_i is the source tweet, c_{ij} is the *j*th response text, and m_i is the number of posts in the event ε_i. Considering the source tweet as the 0th tweet for presentation simply. Then, $\varepsilon_i^c = \{s_i, c_{i1}, ..., c_{im_{i-1}}\}$ can be expressed as $\varepsilon_i^c = \{c_{i0}, c_{i1}, ..., c_{im_{i-1}}\}$. By expressing the propagation order of each post in a propagation event ε_i in terms of time as $\varepsilon_i^t = \{t_{i0}, t_{i1}, ..., t_{im_{i-1}}\}$, with *t* denoting the time, the final rumor propagation event ε_i can be expressed as $\varepsilon_i = \{(c_{i0}, t_{i0}), (c_{i1}, t_{i1}), ..., (c_{im_{i-1}}, t_{im_{i-1}})\}$.

Divide each post ε_i in a propagation event into γ time steps along the time dimension, γ is determined by the number of posts m_i in each propagation event, $\gamma \epsilon \{1, 2, ..., \gamma\}$, for a propagation event ε_i different propagation states at different time steps γ. In the first time step $\varepsilon_{i0} = \{(c_{i0}, t_{i0})\}$, in the second time step $\varepsilon_{i0} = \{(c_{i0}, t_{i0}), (c_{i1}, t_{i1})\}$, so that the event ε_i is finally represented at all time steps as $\varepsilon_i = \{\varepsilon_{i0}, \varepsilon_{i1}, ..., \varepsilon_{i\gamma}\}$.

For the same propagation event ε_i at different time steps, different propagation structure graphs $\mathcal{G}_{ir} =< V_{ir}, E_{ir} >$ are constructed. For easy understanding, the important mathematical notations are listed in Table I.

B. Dynamic-GCN Module

1) Construction of Dynamic Graphs: The idea of constructing the dynamic graph is inspired by Sun et al. [10], and we construct a time-step-based rumor propagation graph, such as the dynamic-GCN part in Fig. 1. The propagation of rumor events develops dynamically according to time, and for the same propagation event, each time step is a network of rumor propagation. With the development of time, the number of these networks increases dynamically, eventually forming a dynamic propagation network. According to the rule that the propagation network changes dynamically over time, a corresponding propagation graph is constructed at each time step, where nodes denote different posts and edges denote the forwarding and replying relationships between these posts.

This section gives the definition of a fine-grained dynamic graph based on time steps: a time step that propagates a fine-grained dynamic pattern of only one node. This is done as follows.

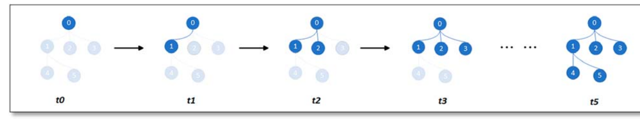

Fig. 3. Fine-grained dynamic graphs based on time steps.

As shown in Fig. 2, for the propagation event ε_i: at the moment T0, only one source post exists (node 0); at moment T1, two response tweets interact with the source post (nodes 1 and 2); at moment T2, only one tweet responds to the source post (node 3); and at moment T3, nodes 4 and 5 respond to node 1 again simultaneously. A coarse-grained dynamic graph state, as shown in Fig. 2, was initially obtained based on this relationship, but some of these moments propagated more than one node at the same time, and a fine-grained segmentation is now required.

In the subsequent computation, in order for the dynamic graph network to be able to interact with features based on time steps in the TES, multiple nodes propagated in the same moment are sorted, resulting in a moment where only one node's state is propagated, as shown in Fig. 3. For nodes 1 and 2, which are propagated at the same time in the moment of t1, although their creation times are the same, they are sorted based on the numbering order of each node. Finally, based on the above node propagation rules and their response and forwarding node numbers, the respective static graphs are constructed at different time steps, and a fine-grained dynamic graph sequence based on the time steps is constructed for the dynamic graph convolution operation in the next section.

2) Dynamic-GCN: This section describes in detail the computation of fine-grained dynamic graphs based on time steps.

For the propagation event, $\varepsilon_i = \{\varepsilon_{i0}, \varepsilon_{i1}, ..., \varepsilon_{i\gamma}\}$ construct snapshots of the propagation graph at different moments $\mathcal{G}_{ir} = \mathcal{G}_{i0}, ..., \mathcal{G}_{ir}$. $H_{ir}^p \in \mathcal{R}^{m_{ir}*F}$ is the feature matrix containing all the nodes of the propagation graph $\mathcal{G}_{ir}$, m_{ir} is the number of nodes of the propagation graph at that moment, and F denotes the features vector of the node. Its adjacency matrix is defined as $A_{ir} \in \mathcal{R}^{m_{ir}*F}$. As in (1) and (2), the dynamic graph convolution layer is defined as

$$H_{ir}^{l+1} = \sigma(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}H_{ir}^{l}W^{l}) \tag{1}$$

where $\hat{D}$ is the matrix, $\hat{A}$ is the adjacency matrix, H_{ir}^l is the feature matrix of all nodes of the previous layer of the propagation graph $\mathcal{G}_{ir}$, W^l is the weight matrix, and σ is the nonlinear activation function.

The operation in (1) is performed on the propagation graph $\mathcal{G}_{ir} = \{\mathcal{G}_{i0}, \mathcal{G}_{i1}, ..., \mathcal{G}_{ir}\}$ for all time steps to obtain the spatial features of all time steps

$$H_{ir} = \{H_{i0}, H_{i1}, ..., H_{ir}\}. \tag{2}$$

At this point, the propagation structure feature of the same propagation event at each moment has been obtained, forming a dynamic feature sequence, which is taken as the input to the temporal event sharing layer (described in detail in Section III-D), at the same time, the dynamic structure features H_{ir} of the last moment is taken as the final output of the dynamic-GCN module, which is used as the input to the classification network (described in detail in Section III-E).

C. Content-LSTM Module

In the previous section, after the fine-grained time-step-based dynamic graph convolution computation, the dynamic structural feature $\{H_{i0}, H_{i1}, ..., H_{ir}\}$, which is a time series-based embedding of structural features, the difference between neighboring time-steps lies in the generation of a certain node (post), which are arranged in a strict temporal order. To make this dynamic feature of dynamic graphs related to content-LSTM in the development of the time step to meet the requirement of combining them based on the time step in TES. In this section, the node diffusion order in Section III-B is used as the arrangement order of the textual content of the corresponding rumor events, and the textual content of the same order is modeled to form a set of time-step-based textual feature sequences. It is used for subsequent computation steps.

Inspired by Wan et al. [14], we use an LSTM network as a baseline model. For the propagation event ε_i, the entire tweets contained are denoted as $X_{i\_\text{content}} = \{X_{i\_0}, X_{i\_1}, ..., X_{i\_n}\} \in \mathcal{R}^{n*300}$, which is used as the input of content-LSTM, as in the following equations:

$$i_t = \sigma(W_{ii}X_{i\_\text{content}\_t} + b_{it} + W_{hi}h_{(t-t)} + b_{hi}) \quad (3)$$

$$f_t = \sigma(W_{if}X_{i\_\text{content}\_t} + b_{if} + W_{hf}h_{(t-1)} + b_{hf}) \quad (4)$$

$$g_t = \tanh(W_{ig}X_{i\_\text{content}\_t} + b_{ig} + W_{hg}h_{(t-1)} + b_{hg}) \quad (5)$$

$$o_t = \sigma(W_{io}X_{i\_\text{content}\_t} + b_{io} + W_{ho}h_{(t-1)} + b_{ho}) \quad (6)$$

$$c_t = f_t * c_{t-} + i_t * g_t \quad (7)$$

$$h_{\text{content}} - t = o_t * \tanh(c_t) \quad (8)$$

where $X_{i\_\text{content}\_t}$ is the input feature vector at a given moment, $h_{(t-1)}$ is the hidden state at the previous moment, and W_* and b_* are the corresponding weight matrix and bias.

At this point, the same propagation event at each moment of the text feature sequence, formed a temporal feature sequence H_{content}, as in (9), notated as

$$h_{\text{content}} = \{h_{\text{content}-0}, h_{\text{content}-1}, ..., h_{\text{content}-t}\}. \quad (9)$$

Note that the time series feature corresponds one-to-one with the dynamic structure sequence feature H_i in (2) in terms of the order of occurrence of the propagation events. This time series feature is taken as the input of the temporal event sharing layer (described in detail in Section III-D); meanwhile, the sequence feature H_{content} of the last moment is taken as the final output of content-LSTM, which is used as the input of the classification network (described in detail in Section III-E).

D. Temporal Event Share Layer

In the above two sections, the dynamic structural feature H_i and the time-series feature H_{content} are obtained based on the rule that the propagation network develops dynamically over time, as well as the temporal order in which the nodes are

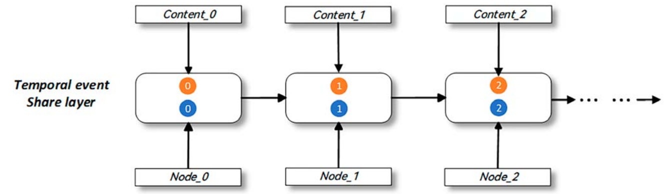

Fig. 4. Structure of TES.

generated, respectively. They are related in the development sequence of propagation events; in other words, for a certain propagation event at the same time stage, we capture its temporal and spatial feature sequences, and their temporality and dynamics are consistent in the postdevelopment process.

In this section, as shown in the middle part of Fig. 1, a temporal event-sharing layer is proposed, which combines dynamic-GCN and content-LSTM with time step as the basic computational unit and shares their features at the same time stage interactively and trains them together. Compared to other multitask learning using hardware parameter sharing [15] as an approach, we additionally implement temporal event sharing, which improves the sharing frequency and efficiency of dynamic-GCN and content-LSTM. The specific computation steps are as follows.

Step 1: As in Fig. 4, at the moment t0, the feature $H_{i0-v} \in \mathcal{R}^{1*F}$ of the node(Node_0) corresponding to the one that is being propagated in the dynamic structural feature $H_{i0} \in \mathcal{R}^{nodes*F}$ in Section III-B is taken, and then the time-series feature $H_{\text{content}-0}$ in Section III-C is taken. Now, the propagation structure features and content features of the corresponding tweets with the number 0 are taken at the same time.

Step 2: Splice them and go through the linear layer mapping as the input of the cell block at the moment t0 in the TES, and go through the gating calculation to get the output of the first time step, as in the following equations:

$$X_{\text{share}-0} = \text{Linear}(\text{Concat}(H_{i0}, H_{\text{content}-0})) \quad (10)$$

$$i_t = \sigma(W_{ii}X_{\text{share}-0} + b_{it} + W_{ht}h_{(t-1)} + b_{hi}) \quad (11)$$

$$f_t = \sigma(W_{if}X_{\text{share}-0} + b_{if} + W_{hf}h_{(t-1)} + b_{hf}) \quad (12)$$

$$g_t = \tanh(W_{ig}X_{\text{share}-0} + b_{ig} + W_{hg}h_{(t-1)} + b_{hg}) \quad (13)$$

$$o_t = \sigma(W_{io}X_{\text{share}-0} + b_{io} + W_{ho}h_{(t-1)} + b_{ho}) \quad (14)$$

$$c_t = f_t * c_{t-1} + i_t * g_t \quad (15)$$

$$H_{\text{share}-0} = o_t * \tanh(c_t). \quad (16)$$

In the above equation, $X_{\text{share}-0}$ is the input of the temporal event sharing layer at the moment t0, W_* and b_* are the weight matrix and bias, and $h_{(t-1)}$ is the hidden layer vector of the previous moment, which is randomly initialized and generated at the moment t0, and $H_{\text{share}-0}$ is the output.

At this point, a similar operation is carried out in the first cell block of the content-LSTM, with the difference that the input is a representation of the text content at the moment t0, and the output is the time series feature $h_{\text{content}-0}$ at the moment t0.

Step 3: Splice $h_{\text{content}-0}$ and $H_{\text{share}-0}$ and undergo linear mapping to update the hidden state of content-LSTM at the moment of t0 as the hidden state of the content-LSTM cell block at the next moment, as in

$$h_{\text{content}-0} = \text{Linear}(\text{Concat}(h_{\text{content}-0}, H_{\text{share}-0})). \tag{17}$$

The computation of the content-LSTM for the next moment is referred to in (3)–(8).

Step 4: Repeat the first step to the third step until all the time steps have been looped to finally get the output of each moment of TES, as in

$$H_{\text{share}} = \{H_{\text{share}-0}, H_{\text{share}-1}, \ldots, H_{\text{share}-t}\}. \tag{18}$$

Note that the TES is continuously interacting with the information in dynamic-GCN and content-LSTM in chronological order during the computation process, thus realizing temporal event sharing.

E. Attentional Mechanisms and Classifier

In the previous section, the computation process has been completed and the feature representation after feature sharing has been obtained through TES, which are dynamic-GCN, content-LSTM, and TES layers: H_{ir}, H_{content}, and H_{share}.

To understand which features are more important for prediction across different time series and across different tasks, this section updates them using a multihead attention mechanism [31]. Rumor detection efforts in LSTM networks typically use the latent feature representation of the final time series as the predicted feature vector [14], since the sequence data are computed through a gate mechanism, and the output of the last gate also contains all of the previous gating computed hidden states. Therefore, we also use the features of the last time series for computation. As in the right part of Fig. 1, we obtain the convolutional features H_{it} of the dynamic graph convolutional network at the final moment for subsequent computation. The details of the implementation are as follows.

In the first step, $H_{\text{share}-t}$ is spliced with $H_{\text{content}-t}$ and H_{i-t}, respectively, and mapped by a linear layer to obtain the inputs X_1 and X_2 of the attention layer as in the following equations:

$$X_1 = \text{Linear}(\text{Concat}(H_{\text{content}-t}, H_{\text{share}-t})) \tag{19}$$

$$X_2 = \text{Linear}(\text{Concat}(H_{i-t}, H_{\text{share}-t})). \tag{20}$$

In the second step, X_1 and X_2 are linearly transformed into three projections with the same dimensions as itself (query Q; key K; and value V), which are used as inputs to the attention layer, as in the following equations:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_n)W^o \tag{21}$$

$$\text{head}_i = \text{Attention}(XW_i^Q, XW_i^K, XW_i^V) \tag{22}$$

$$\text{Attention}(O, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}V\right). \tag{23}$$

MultiHead(Q, K, V) is the output of the attention layer, Q, K, and V denote XW_i^Q, XW_i^K, XW_i^V, and W^o is the weight matrix, W_i is the initialization of the different multimatrices, d_k is the dimension of *K*, scaling the output when *K* is larger.

In the third step, the outputs of the attention layer are spliced with $H_{\text{share}-t}$ and H_{i-t}, and go through a linear mapping, which is used to update them, as in the following equations:

$$H_{\text{content}-t} = \text{Linear}(\text{Concat}(H_{\text{share}-t}, H_{\text{content}-t})) \tag{24}$$

$$H_{i-t} = \text{Linear}(\text{Concat}(H_{\text{share}-t}, H_{i-t})). \tag{25}$$

In the fourth step, the updated, final event feature representations $H_{\text{content}-t}$,H_{i-t} for prediction are obtained, and the corresponding fully connected and activation layers are designed for them, respectively, as in the following equations:

$$Y_c = \text{softmax}(W_c H_{\text{content}-t} + b_c) \tag{26}$$

$$Y_{DG} = \text{softmax}(W_{DG} H_{i-t} + b_{DG}). \tag{27}$$

W_* and b_* are their respective corresponding weights and bias parameters, and finally, the cross entropy loss function is used as the classification loss as in the following equations:

$$L_c = -\sum_i Y_i \log Y_C \tag{28}$$

$$L_{DG} = -\sum_i Y_i \log Y_{DG}. \tag{29}$$

Y_i is the true label of the *i*th event. For training, the two loss functions are assigned corresponding coefficients and summed to find the total loss function for backpropagation as in (30). $W1$ and $W2$ represent the parameters of the two tasks

$$\text{Loss} = W1 * L_C + W2 * L_{DG}. \tag{30}$$

TABLE II
STATISTICS OF DATASETS

Statistic	WEIBO	PHEME
Posts/events	4663	5473
Non − rumors	2312	1996
Rumors	2351	3477
Avg posts/event	452	20
Max posts/event	44 764	346
Min posts/event	2	3

IV. EXPERIMENTS

This section evaluates the superiority of our method by comparing the explored method with other methods based on several publicly available datasets. In addition, ablation experiments and other complementary experiments are performed to illustrate the effectiveness of the various components of our method.

A. Datasets

To investigate the content and propagation patterns of fake news, the WEIBO dataset [5] and PHEME dataset [15] are selected. After removing some samples, the dataset is shown in Table II, which contains both false and true information from both sites. Each event contains several response tweets with comments and retweets, and all have a label, rumor, or nonrumor. For the WEIBO dataset, due to the large average number of posts per rumor event, the WEIBO dataset is divided into training, validation, and testing sets in the ratio of 6:2:2, and for the PHEME dataset in the ratio of 8:1:1, taking into account the balance of the sample distribution. To compare the detection results with other models to validate the effectiveness of the model, the experiments used four evaluation metrics, namely, accuracy, precision, recall, and F1 score. The F1 score was used to evaluate the performance of the model. The calculation formulas are shown in the following equations, respectively, where TP denotes a true-positive rate, TN denotes a true-negative rate, FP denotes a false-positive rate, and FN denotes a false-negative rate:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{31}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{32}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{33}$$

$$F1 = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{34}$$

B. Comparison Methods

We compare with the following baselines.

DTC [32]: a rumor detection method using Decision Tree classifier with hand-crafted features.

SVM-RBF [33]: an SVM model with RBF kernel using handcrafted features based on the overall statistics of the posts for classification.

SVM-TS [34]: A time-series structure-based SVM classifier that extracts features by modeling the content, users, and propagation patterns of rumors as well as their impact on society.

RvNN [5]: a method that uses a tree-structured recursive neural network to extract the high-level representation from both text contents and propagation structures, which reinforces or weakens the stance of the node by judging whether to respond to the node.

TGNF [9]: a temporally evolving graph neural network model for fake news detection fusing interactive and temporal features.

Bi-GCN [3]: a GCN-based rumor detection model utilizing the bidirectional propagation structure to obtain the propagation and dispersion features.

STS-NN [25]: a spatial–temporal structure neural network to treat the spatial structure and the temporal structure as a whole to model the message propagation for rumor detection.

DDGCN [10]: a dual-dynamic graph convolutional network to model the spatial structure, temporal structure, external knowledge, and text information in one unified framework.

C. Parameter Settings

For all comparison methods, the default settings from the corresponding articles were used. All hyperparameters are specified in Table III. For the methods in this study, the parameters were optimized using the Adma algorithm. The learning rate is 1e-3, and the batch sizes are 32 and 128. For the embedding representation, this article uses the spaCY open source AIP pretraining model TRANSFORMER to process each sentence to get the embedding representation with 300 dimensions.

TABLE III
MAIN HYPERPARAMETER SETTINGS

Datasets	Hyperparameter	
PHEME	Average subgraph nodes	20
	Max subgraph nodes	346
	Time steps	20
	Hidden Dim	128
	Batch size	128
WEIBO	Average subgraph nodes	40
	Max subgraph nodes	44 764
	Time steps	452
	Hidden Dim	128
	Batch size	32

TABLE IV
RUMOR DETECTION RESULTS ON THE WEIBO DATASET

Method	Acc	Pre	Recall	F1
DTC	0.858	0.834	0.822	0.857
SVM-RBF*	0.899	0.938	0.846	0.889
SVM-TS	0.885	0.950	0.932	0.938
$RvNN_{BU}(ACL18)$	0.908	0.912	0.897	0.905
STS-NN	0.913	0.902	0.898	0.900
Bi-GCN(AAAI20)	0.919	0.918	0.916	0.913
TGNF(IPM21)*	0.968	0.974	0.960	0.967
DDGCN(AAAI22)*	0.948	0.953	0.948	0.950
ours	**0.974**	**0.978**	**0.970**	**0.975**

Note: The bold entries are the relative best values.

TABLE V
RUMOR DETECTION RESULTS ON THE PHEME DATASET

Method	Acc	Pre	Recall	F1
DTC	0.581	0.659	0.652	0.656
SVM-RBF	0.602	0.875	0.431	0.577
SVM-TS	0.651	0.642	0.686	0.663
$RvNN_{BU}(ACL18)$	0.789	0788	0.788	0.789
STS-NN	0.819	0.816	0.791	0.800
Bi-GCN(AAAI20)	0.847	0.840	0.834	0.835
TGNF(IPM21)*	0.848	**0.892**	0.861	0.877
DDGCN(AAAI22)*	0.855	0.846	0.841	0.844
ours	**0.869**	0.878	**0.877**	**0.877**

Note: The bold entries are the relative best values.

D. Experimental Results and Analyses

The proposed model is compared with the baseline method mentioned in Section IV-B, Tables IV and V show the performance of the compared methods, where the first group is the machine learning method and the second group is the deep learning approach. Fig. 5 also shows more visually the comparison of the proposed method in this article with other baselines using F1 scores. We ran five runs based on the default settings in the original paper and reported the average results here, "*" Experimental results were obtained directly from the original paper.

From the above table, the following analysis can be obtained.

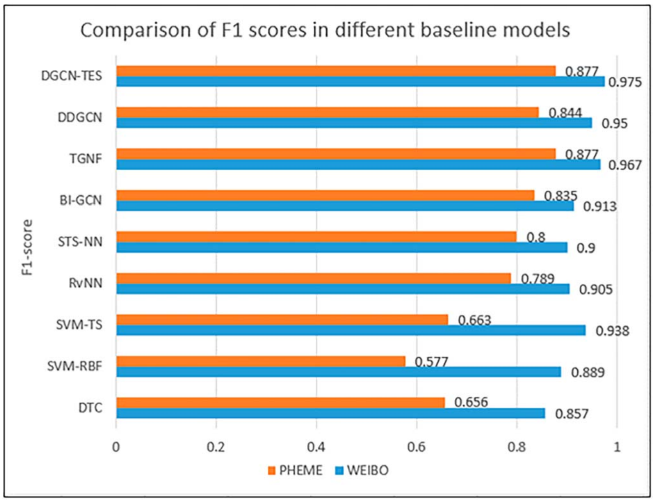

Fig. 5. Comparison of F1-score in different baseline models.

The machine learning methods (DTC, SVM-RBF, and SVM-TS) have the worst performance scalar, none of which is better than the second group. This is due to the fact that the information extracted by feature engineering is often a latent representation of the data and does not capture deeper feature information, whereas deep learning is able to learn high-level semantic information about the data. However, a comparison of the first group of methods with each other shows that SVM-TS outperforms the other two methods due to the development of a dynamic series of time-structured models that are able to explore the patterns of various social contextual features over time, and feature design that incorporates social contextual information.

In the second group, the RvNN that utilizes the propagation tree model performs better than SVM-TS, the accuracy is 2.59% and 21.19% higher on WEIBO and PHEME datasets, respectively, this is due to the fact that it models the rumor events as a propagation tree, the leaf nodes will influence and learn each other's information; however, the gap on the WEIBO dataset is relatively close, which may be due to the fact that the WEIBO dataset is caused by the higher average number of posts when the average number of posts of events is higher, the propagation tree constructed is richer, and the structural features captured are more obvious. STS-NN proposes a spatiotemporal structural neural network, which models the spatial and temporal structure as a whole for message propagation for rumor detection and does not construct a graph structure to capture the spatial features, so its accuracy is 0.5% higher than that of the RvNN by 0.4%, 3.8%, respectively, which is between RvNN and Bi-GCN.

Bi-GCN that utilizes the propagation graph structure has better performance, it outperforms RvNN and STS-NN by 1.2%, 7.35%, and 0.65%, 3.41% on the two datasets, respectively, thanks to its use of static graph related algorithms. It learns the embedding representation of graphs from a bidirectional point of view, designing a bidirectional convolutional algorithm to learn better features. But it does not use dynamic ideas.

TABLE VI
RESULTS OF COMPARISON AMONG DIFFERENT VARIANTS

Datasets	Method	acc	pre	recall	F1
WEIBO	W/o GCN	0.906	0.902	0.902	0.902
	W/o TES	0.952	0.961	0.962	0.961
	W/o dynamic	0.961	0.961	0.969	0.965
	DGCN-TES	**0.974**	**0.978**	**0.970**	**0.975**
PHEME	W/o GCN	0.812	0.812	0.812	0.813
	W/o TES	0.828	0.823	0.826	0.824
	W/o dynamic	0.838	0.829	0.840	0.834
	DGCN-TES	**0.869**	**0.878**	**0.877**	**0.877**

Note: The bold entries are the relative best values.

Algorithms using dynamic ideas will have better results, TGNF models the temporal interaction events of nodes from the dynamic evolution perspective, and the accuracy is 5.33% and 0.1% higher than Bi-GCN, respectively. DDGCN is 0.82% more accurate than TGNF on the PHEME dataset due to a kind of dynamic graph convolutional network.

Overall, DGCN-TES has the best performance in most of the metrics, and the fine-grained time-step-based dynamic graph network we designed is able to capture dynamic sequence features with more delicate transitions, get stronger feature representations from interaction training in TES, and utilize the attention mechanism, which is able to act as filtering and accelerate model convergence, and thus outperforms Bi-GCN by 5.98% and 2.59%, respectively, and DDGCN by 2.74% and 1.63%, respectively, than Bi-GCN using the static graph algorithm.

The method DGCN-TES proposed in this study achieves better performance, thanks to the fact that we model and interact with information about the dynamic spatial structure and temporal text content. Specifically, there are the following advantages that other methods do not do:

1) In this article, we constructed a time-step level, fine-grained dynamic graph convolutional layer based on the time-step level, where a snapshot of the dynamic graph is created for each extension of a node. More delicate and fine-grained features are captured in the dynamic relationship model.
2) Using the form of multitasking to combine the sequence data with the dynamic relationship, in the design, it is done that the time in the sequence data corresponds to the time in the dynamic relationship, dynamic-GCN and content-LSTM are combined at the time-step level, which makes both of them more powerful and more effective in sharing interaction.

E. Ablation Experiment

To further investigate the impact of the individual parts in this method, this section conducts ablation experiments using the comparison strategy shown in Table VI. The model is compared to the network with each part removed.

The following combinations are defined to experimentally demonstrate the effect of each of our proposed components.

1) *W/o TES:* When the model does not use TES, the overall architecture is content-LSTM and dynamic-GCN as well

as classifiers. Despite the lack of TES, the overall form of multitasking remains.

2) *W/o dynamic:* When the model does not use DGNN, the overall architecture is content-LSTM, GCN, TES, Attention mechanism, and Classifier.
3) *W/o GCN:* When the model does not use a graph-convolutional network, the TES cannot be shared due to the lack of a graph-convolutional layer, leaving the network with only the content-LSTM and the classifier.

The following analysis can be obtained from Table VI.

W/o GCN: This effect is the worst, due to the fact that the network loses objects that can be shared, leaving only the content-LSTM layer for detection, and is unable to capture spatial features as well as dynamic features.

W/o TES: At this time, the network is still in the mode of multitasking, but due to the loss of TES, the sharing strategy is changed, and relying on backpropagation alone for parameter sharing is limited and insufficient to achieve the best performance, but the accuracy is 5.07% and 1.97% higher w/o GCN strategy, respectively, which shows the important impact of the time-step-based TES.

W/o dynamic: This strategy is the best in addition to DGCN-TES and the accuracy is higher than w/o TES by 0.94% and 1.2%, respectively, mainly because this approach basically realizes multitasking combined with temporal event sharing. The two tasks are able to capture temporal features, spatial features together, and at the same time, because of the participation of the attention mechanism, the performance is better than other strategies. However, due to the use of static graphs, it ignores the dynamics of rumor events in the evolution process.

By comparing with other strategies, DGCN-TES is the optimal method for each combined strategy. The model can fully capture the content features of sequence data, the structural features of rumor propagation events, and the dynamic feature of the propagation process, combining them in a multitask fashion and designing a time-series event-sharing layer so that the two tasks can better learn from each other through fine-grained time steps. Finally, the attention mechanism is utilized to focus on the features that have an important effect on prediction between different tasks and different time steps and ignore those less important information so as to improve the model effect.

In summary, the following conclusion is drawn: the DGCN-TES is the best performer, which proves that the idea of combining textual content information in the form of multitasking with a dynamic dissemination structure, and then utilizing temporal event sharing to enhance the interaction between the multitasks, is effective and has important implications.

F. Compare and Contrast Different Time Granularity Dynamics Graph

At the early stage of the experiment, we had a conjecture: how does the coarseness of the time-step division granularity and the number of static graph snapshots constructed affect the experiment? We believe that the finer the granularity and the more static graph snapshots are constructed, the more delicate and more effective the transition of the generated dynamic graph features will be, which will lead to a better overall performance of the model.

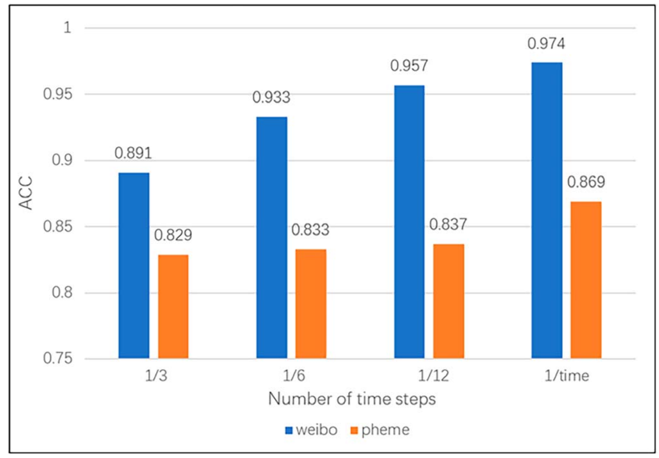

Fig. 6. Different time particle size comparison experiments.

In this section, to investigate how dynamic graphs with different granularities affect network performance, this issue is investigated by actively controlling the time interval for generating snapshots of static graphs. Experiments were conducted on the PHEME dataset and the WEIBO dataset, following several sets of different time granularity strategies. The results are shown in Fig. 6, where the horizontal axis indicates the number of set time steps relative to the average event propagation length, and the vertical axis indicates the accuracy.

From Fig. 6, it is easy to find that when the time interval is smaller, a static graph snapshot with a smoother transition will be constructed; and thus, the captured structural features are more delicate and more orderly. More importantly, the smaller the interval, the better the interaction between the dynamic graph network layer and the timing event sharing layer, the closer the information interaction between multiple tasks, and the better the performance of the model.

Coarser granularity and worst results when the time step is split into three (1/3). However, comparing the (1/6) time steps, the accuracy of the WEIBO and PHEME datasets is improved by 4.71% and 0.48%, respectively, and this gap is not very obvious in the PHEME dataset, which may be due to the fact that the average number of posts of the propagation events in the PHEME dataset is less, and the transition difference of the dynamical graph features of the different time steps is closer to each other, which ultimately leads to the results of the different time granularity is closer, and the accuracy increases steadily as the time granularity increases.

Therefore, it was concluded as follows.

1) The model performs better when the temporal granularity is finer, the transitions of the captured dynamical graph features are more delicate, and the interactions in the TES are closer.
2) The average number of posts of a propagation event will directly determine the number of time steps, and when the average number of posts of a propagation event is small,

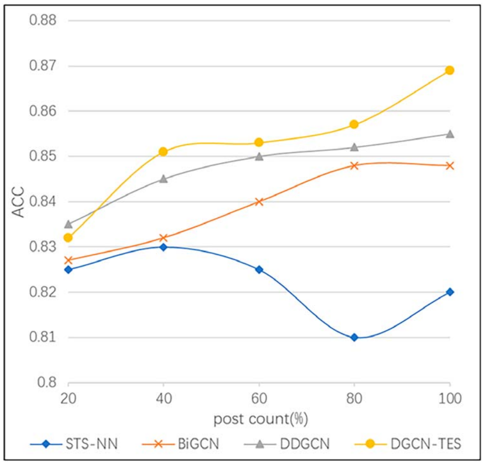

Fig. 7. Early detection on the PHEME dataset.

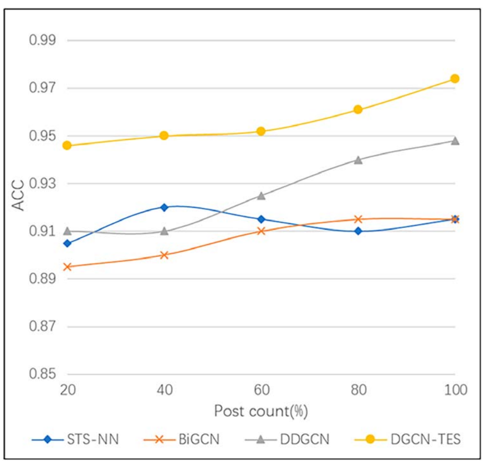

Fig. 8. Early detection on the WEIBO dataset.

the gap between DGCN-TES at different time granularities is closer.

G. Early Detection

Early rumor detection is one of the important goals of rumor detection, a task aimed at identifying rumor events in the early stages of their propagation, and this means of assessing the quality of rumor detection methods is another important reference point.

To construct an early rumor detection task based on DGCN-TES, this experiment on one hand controls the number of nodes generated by inputting the rumor propagation graph into the dynamic graph convolutional layer, and on the other hand, intercepts the text sequence data corresponding to this time, and represents the different periods of rumor events by reconstructing the rumor propagation events. Specifically, this experiment deletes the corresponding tweets posted after a fixed time step to reconstruct the propagation event dataset of the model, and after that, in the same way, we take the sequential text content corresponding to the time period and construct the dynamic propagation graph, and then evaluate the performance of the rumor detection model in different periods. As in Figs. 7 and 8, the differences between DGCN-TES and other approaches on the PHEME dataset and WEIBO dataset are shown respectively.

In terms of the dataset, the WEIBO dataset performs better than the other methods in any period of time, and the accuracy steadily increases as the early detection phase develops. This may be due to the fact that the WEIBO dataset propagates events with a larger average number of posts, the dynamic excess of the time step is very delicate, and the learned dynamic structural features are able to interact with the temporal features at a higher frequency. From this perspective, it can explain why the performance of the PHEME dataset is more similar to the DDGCN method in the early detection phase from 20% to 60%, and then pulls away from 80% to 100%.

From the results, the proposed method DGCN-TES achieves relatively high accuracy in the early stage of rumor event propagation, in addition, STS-NN is the worst in early detection, this is because all other networks adopt graph network related operation, which suggests that the graph network is more conducive to deal with early detection. As for the methods using graph networks, the best results are achieved by using dynamic graph networks, and our curves are more similar to DDGCN in the early stage of rumor propagation and superior to DDGCN in the subsequent stage of rumor propagation, which is due to the fact that our dynamic temporal granularity is more fine-grained, and the more delicate dynamic graph structure demonstrates a more superior performance in both early detection and long-term rumor detection.

H. Exploring the Impact of Different Variants of the Model

In the above-mentioned work, this article combines dynamic-GCN and content-LSTM in the form of multitasking, although the two base components are obtained by modifying the GCN and LSTM networks, the most basic original model is used and no related variant network is used. However, in the field of NLP research, several studies have carried out richer and more effective work using variant networks, e.g., Pandey et al. [35] used an LSTM network based on an attention mechanism to recognize sarcastic statements, and Pandey and Singh [36] proposed a BERT-based bidirectional encoder model that stacks LSTM networks to perform recognition prediction.

Relevant research has all shown that the variant model obtained by improving on the original baseline model has better performance, therefore, to better explore how much different variant models affect DGCN-TES, in this section, this article replaces the baseline components in DGCN-TES and carries out experiments. Specifically, two variant network experimental scenarios are carried out in this section as follows.

1) Using attention-based LSTM network instead of content-LSTM component in DGCN-TES.
2) Using bi-LSTM network instead of TES component in DGCN-TES

The experimental results are shown in Figs. 9 and 10.

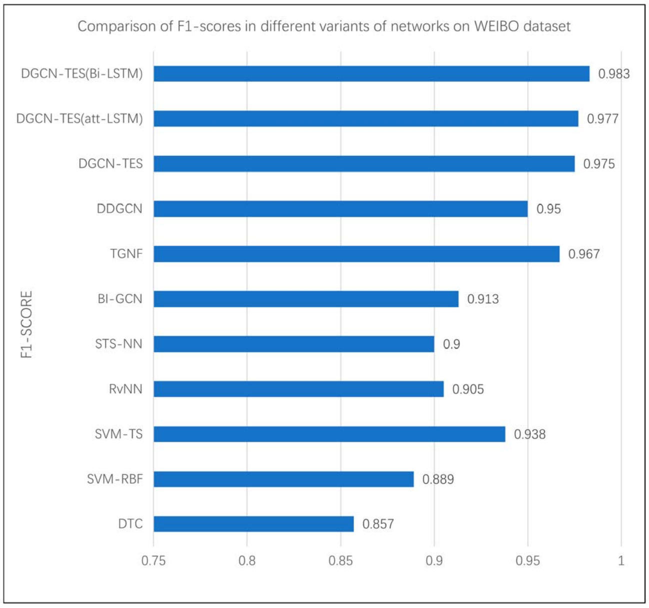

Fig. 9. Comparison of F1-score in different variants of networks on WEIBO dataset.

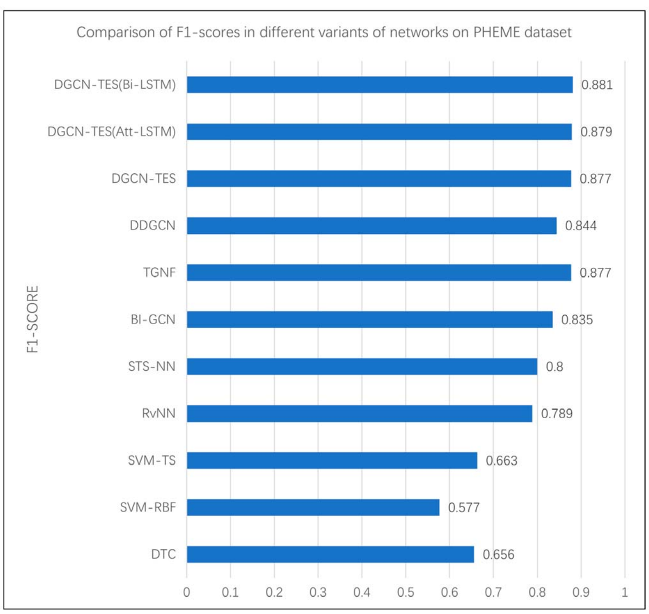

Fig. 10. Comparison of F1-score in different variants of networks on PHEME dataset.

It is clearly visible through Fig. 9 and Fig. 10 that the experimental results of using the attention mechanism-based LSTM instead of content-LSTM are better than DGCN-TES, and by comparing the F1 scores on the WEIBO and PHEME datasets, it can be found that the DGCN-TES (Att-LSTM) exceeds the DGCN-TES by 0.21% and 0.23%, which is mainly because when dealing with sequential text data, the attention mechanism is able to focus on feature information in the context that is more beneficial for rumor detection, thus learning better feature representations, which ultimately enhances the overall performance of the model. The variant network scheme using Bi-LSTM instead of content-LSTM is more effective, exceeding 0.82% and 0.46% on WEIBO and PHEME datasets, which is mainly due to the fact that bi-LSTM uses bidirectional sequential structure, which is not only focuses on sequential features from top to the bottom but also focuses on the effective information from the bottom to the top when dealing with the temporal sequential text information. This way of processing is more dynamic and also makes the dynamic feature interaction between the "previous time step" and "next time step" of the dynamic graph in the dynamic-GCN component more effective, thus showing better performance.

More in-depth exploration using variant networks can further improve the performance of the model, but the focus of this article is on exploring dynamic patterns in the rumor propagation process, not in this context (e.g., using the GAT network instead of the GCN network in dynamic-GCN; and using the BERT instead of content-LSTM), and therefore no more in-depth experiments are conducted using other variant networks in this section.

V. Conclusion

In this article, we propose a network named DGCN-TES, which models dynamic spatial structure, temporal structure, and textual content information under a unified architecture. DGCN-TES contains a fine-grained dynamic graph convolutional layer based on temporal sequences and a textual content-based LSTM layer for processing spatial and content information over time. In addition, we designed a temporal event-sharing module for sharing the interaction of features of different temporal in the two tasks to learn from each other during the training process. Finally, the attention mechanism is utilized to focus on the features that are beneficial for prediction between different tasks and time series to improve the network's effectiveness. Our experiments get better feedback on two public datasets, and DGCN-TES outperforms other methods. Finally, we did additional supplementary experiments to explore the effect of different temporal granularities on the performance of dynamic networks and found that more delicate temporal granularities are more suitable for dynamic networks. An early detection study was also conducted, and the DDGCN-TES performance was also superior in comparison with other methods. An extended research is also done on the effect of different variant networks on the model, and by using different variant networks for experiments, it is found that a more suitable variant network can bring better performance improvement.

In the future, we will further explore the properties of rumor-spreading events in dynamic networks, and process the multimodal information in rumor-spreading events for better network performance.

References

[1] N. DiFonzo and P. Bordia, "Rumor psychology: Social and organizational approaches." *Amer. Psychol. Assoc.*, 2007.

[2] J. Cohen and D. Normile, "New SARS-like virus in China triggers alarm," *Amer. Assoc. Adv. Sci.*, vol. 367, no. 6475, pp. 234–235, 2020.

[3] T. Bian et al., "Rumor detection on social media with bi-directional graph convolutional networks," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 1, 2020, pp. 549–556.

[4] N. Bai, F. Meng, X. Rui, and Z. Wang, "Rumor detection based on a source-replies conversation tree convolutional neural net," *Computing*, vol. 104, pp. 1–17, Jan. 2022.

[5] J. Ma, W. Gao, and K.-F. Wong, "Rumor detection on Twitter with tree-structured recursive neural networks," in *Proc. 56th Ann. Meeting Association for Comp. Linguistics*, Association for Computational Linguistics, 2018, pp. 1980–1989.

[6] V. Vaibhav, R. M. Annasamy, and E. Hovy, "Do sentence interactions matter? Leveraging sentence level representations for fake news classification," 2019, *arXiv:1910.12203*.

[7] X. Zhou and R. Zafarani, "A survey of fake news: Fundamental theories, detection methods, and opportunities," *ACM Comput. Surv.*, vol. 53, no. 5, pp. 1–40, 2020.

[8] C. Song, Y. Teng, Y. Zhu, S. Wei, and B. Wu, "Dynamic graph neural network for fake news detection," *Neurocomputing*, vol. 505, pp. 362–374, Sep. 2022.

[9] C. Song, K. Shu, and B. Wu, "Temporally evolving graph neural network for fake news detection," *Inf. Process. Manage.*, vol. 58, no. 6, 2021, Art. no. 102712.

[10] M. Sun, X. Zhang, J. Zheng, and G. Ma, "DDGCN: Dual dynamic graph convolutional networks for rumor detection on social media," in *Proc. AAAI Conf. Artif. Intell.*, vol. 36, no. 4, 2022, pp. 4611–4619.

[11] J. Ma, W. Gao, P. Mitra, S. Kwon, and B. J. Jansen, "Detecting rumors from microblogs with recurrent neural networks," in *Proc. 25th Int. Joint Conf. Artif. Intell.*, ser. IJCAI'16, 2016, pp. 3818–3824.

[12] T. Ahmad, M. S. Faisal, A. Rizwan, R. Alkanhel, P. W. Khan, and A. Muthanna, "Efficient fake news detection mechanism using enhanced deep learning model," *Appl. Sci.*, vol. 12, no. 3, 2022, Art. no. 1743.

[13] J. Ma, W. Gao, and K.-F. Wong, "Detect rumor and stance jointly by neural multi-task learning," in *Proc. Companion Web Conf.*, 2018, pp. 585–593.

[14] S. Wan, B. Tang, F. Dong, M. Wang, and G. Yang, "A writing style-based multi-task model with the hierarchical attention for rumor detection," *Int. J. Mach. Learn. Cybern.*, vol. 14, no. 11, 2023, pp. 1–16.

[15] E. Kochkina, M. Liakata, and A. Zubiaga, "All-in-one: Multi-task learning for rumour verification," 2018, *arXiv:1806.03713*.

[16] J. P. Singh, A. Kumar, N. P. Rana, and Y. K. Dwivedi, "Attention-based LSTM network for rumor veracity estimation of tweets," *Inf. Syst. Frontiers*, vol. 24, pp. 1–16, Aug. 2020.

[17] J. Ma, W. Gao, and K.-F. Wong, "Detect rumors on Twitter by promoting information campaigns with generative adversarial learning," in *Proc. World Wide Web Conf.*, 2019, pp. 3049–3055.

[18] M. Cheng, Y. Li, S. Nazarian, and P. Bogdan, "From rumor to genetic mutation detection with explanations: A GAN approach," *Sci. Rep.*, vol. 11, no. 1, 2021, Art. no. 5861.

[19] S. Vosoughi, D. Roy, and S. Aral, "The spread of true and false news online," *Science*, vol. 359, no. 6380, pp. 1146–1151, 2018.

[20] F. Jin, E. Dougherty, P. Saraf, Y. Cao, and N. Ramakrishnan, "Epidemiological modeling of news and rumors on Twitter," in *Proc. 7th Workshop Social Netw. Mining Anal.*, 2013, pp. 1–9.

[21] P. Zhang, H. Ran, C. Jia, X. Li, and X. Han, "A lightweight propagation path aggregating network with neural topic model for rumor detection," *Neurocomputing*, vol. 458, pp. 468–477, Oct. 2021.

[22] A. Silva, Y. Han, L. Luo, S. Karunasekera, and C. Leckie, "Propagation2Vec: Embedding partial propagation networks for explainable fake news early detection," *Inf. Process. & Manage.*, vol. 58, no. 5, 2021, Art. no. 102618.

[23] Z. Wu, D. Pi, J. Chen, M. Xie, and J. Cao, "Rumor detection based on propagation graph neural network with attention mechanism," *Expert Syst. Appl.*, vol. 158, 2020, Art. no. 113595.

[24] X. Yang, Y. Lyu, T. Tian, Y. Liu, Y. Liu, and X. Zhang, "Rumor detection on social media with graph structured adversarial learning," in *Proc. 29th Int. Conf. Int. Joint Conf. Artif. Intell.*, 2021, pp. 1417–1423.

[25] Q. Huang, C. Zhou, J. Wu, L. Liu, and B. Wang, "Deep spatial–temporal structure learning for rumor detection on Twitter," *Neural Comput. Appl.*, vol. 35, no. 18, pp. 1–11, 2020.

[26] Y. Zhang and Q. Yang, "A survey on multi-task learning," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 12, pp. 5586–5609, Dec. 2022.

[27] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proc. 25th Int. Conf. Mach. Learn.*, 2008, pp. 160–167.

[28] L. Wu, Y. Rao, H. Jin, A. Nazir, and L. Sun, "Different absorption from the same sharing: Sifted multi-task learning for fake news detection," 2019, *arXiv:1909.01720*.

[29] N. Bai, F. Meng, X. Rui, and Z. Wang, "A multi-task attention tree neural net for stance classification and rumor veracity detection," *Appl. Intell.*, vol. 53, no. 9, pp. 10715–10725, 2023.

[30] M. Cheng, S. Nazarian, and P. Bogdan, "VRoC: Variational autoencoder-aided multi-task rumor classifier based on text," in *Proc. Web Conf.*, 2020, pp. 2892–2898.

[31] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, no. 1, pp. 261–272, 2017.

[32] C. Castillo, M. Mendoza, and B. Poblete, "Information credibility on Twitter," in *Proc. 20th Int. Conf. World Wide Web*, 2011, pp. 675–684.

[33] F. Yang, Y. Liu, X. Yu, and M. Yang, "Automatic detection of rumor on Sina Weibo," in *Proc. ACM SIGKDD Workshop Mining Data Semantics*, 2012, pp. 1–7.

[34] J. Ma, W. Gao, Z. Wei, Y. Lu, and K.-F. Wong, "Detect rumors using time series of social context information on microblogging websites," in *Proc. 24th ACM Int. Conf. Inf. Knowl. Manage.*, 2015, pp. 1751–1754.

[35] R. Pandey, A. Kumar, J. P. Singh, and S. Tripathi, "Hybrid attention-based long short-term memory network for sarcasm identification," *Appl. Soft Comput.*, vol. 106, 2021, Art. no. 107348.

[36] R. Pandey and J. P. Singh, "BERT-LSTM model for sarcasm detection in code-mixed social media post," *J. Intell. Inf. Syst.*, vol. 60, no. 1, pp. 235–254, 2023.