SWAM: adaptive Sliding Window and memory-augmented Attention Model for Rumor Detection

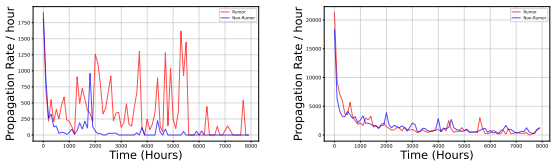
Anonymous ACL submission

Abstract

Detecting rumors on social media has become a critical task in combating misinformation. Existing propagation-based rumor detection methods often focus on the static propagation graph, overlooking that rumor propagation is inherently dynamic and incremental in the real world. So recent propagation-based rumor detection models attempt to use the dynamic graph that is associated with coarse-grained temporal information. However, these methods fail to capture the long-term time dependency and detailed temporal features of propagation. To address these issues, we propose a novel adaptive Sliding Window and memory-augmented Attention Model (SWAM) for rumor detection. The adaptive sliding window divides the sequence of posts into consecutive disjoint windows based on the propagation rate of nodes. We also propose a memory-augmented attention to capture the long-term dependency and nodes' depths in the propagation graph. Multi-head attention mechanism is applied between nodes in the memorybank and incremental nodes to iteratively update the memorybank, and the depth information of nodes is also considered. Finally, the propagation features of nodes in the memorybank are utilized for rumor detection. Experimental results on two public real-world datasets demonstrate the effectiveness of our model compared with the state-of-the-art baselines.

1 Introduction

Social media has become an essential platform for daily communication and information sharing. With the widespread use of social media, an increasing number of people share a variety of posts online. However, this also facilitates the rapid spread of rumors, which have a detrimental impact on public trust and societal discourse. Therefore, rumors detection on social media is crucial to mitigating their harmful effects.



(a) TWITTER dataset (b) DRWeibo dataset

Figure 1: The propagation rate over time on two datasets.

Rumor detection mainly relies on machine learning methods based on feature engineering to identify rumors (Castillo et al., 2011; Yang et al., 2012; Feng et al., 2012; Kwon et al., 2013). With the development of deep learning, various neural networks, such as Recurrent Neural Networks (RNNs), Convolutional Neural Networks (CNNs), and Graph Neural Networks (GNNs) have been proposed for the automatic rumor detection (Ma et al., 2016, 2018; Liu and Wu, 2018; Bian et al., 2020). For GNN-based methods, events are usually modeled as propagation graphs to capture the characteristics of the spreading process. Some studies (Bian et al., 2020; Nguyen et al., 2020; Min et al., 2022; Tian et al., 2022) have explored the static propagation graph of events and achieved superior detection performance. They consider the static graph structure of the final state of event propagation and ignore the temporal dynamics of the propagation.

Recent studies (Lao et al., 2021; Chang et al., 2024; Choi et al., 2021; Sun et al., 2022a; Xu et al., 2024) have explored the temporal dynamics of events and proposed the dynamic graphs to model the spread of events on social media. These dynamic propagation-based rumor detection methods typically divide rumor events on social media into multiple time snapshots, where each snapshot represents the propagation state at a fixed time point. Those methods, usually built by GNNs, emphasize the transformation and aggregation of nodes'

features but fail to capture the detailed temporal features of propagation, such as speed, depth, and breadth. A metric *propagation rate* is defined to measure the speed of event propagation. Figure 1 illustrates the propagation rates of nodes at various timestamps on both the TWITTER and DRWeibo datasets. It is observed that the propagation speeds of rumor and non-rumor vary at different stages. In the early stages, both rumor and non-rumor exhibit a sharply fluctuated speed. Then when the event's propagation continues, the speed of non-rumors becomes stable while rumors' speeds still change. In addition, dividing the propagation process into fixed snapshots makes it challenging to model the event spread over a long time effectively.

To address these issues, we define the *propagation rate* to consider the propagation speed of an event and propose adaptive sliding windows to adjust the number of nodes in consecutive windows. We also propose a novel memory-augmented attention to capture the node's depth and long-term posts' interactions in the propagation graph. Specifically, we first define an initial window and then dynamically adjust the number of nodes in the subsequent window based on the propagation rate of the nodes within the previous window. We take the initial window as a memorybank and update it incrementally with nodes from subsequent windows. Furthermore, we integrate hierarchical information from the propagation structure to enhance the utilization of structural relationships between nodes. Finally, we leverage the representations from the memorybank to enhance the effectiveness of the proposed model. The main contributions of this paper can be summarized as follows:

- We propose an adaptive sliding window to dynamically allocate nodes to consecutive windows, and the number of nodes in each window is determined by the propagation rate of the previous window's nodes.
- We propose the adaptive Sliding Window and memory-augmented Attention Model (SWAM) for rumor detection. Memory-augmented attention saves initial nodes in a memorybank and incrementally updates this memorybank with new nodes using multi-head attention. The depths of nodes are also considered to capture the interactions between posts in the propagation graph.
- We conduct extensive experiments on two real-world datasets to demonstrate the effectiveness of our proposed model on rumor detection.

2 Related Work

2.1 Rumor Detection

Traditional rumor detection methods rely on handcrafted feature engineering to extract rumor features (Castillo et al., 2011; Feng et al., 2012). With the advancement of deep learning, numerous neural network-based methods have been introduced for rumor detection. These approaches can be broadly categorized into content-based methods (Ma et al., 2019; Dun et al., 2021; Xu et al., 2022; Nguyen et al., 2020; Dou et al., 2021; Tian et al., 2022; Min et al., 2022) and propagation structure-based methods (Bian et al., 2020; He et al., 2021; Wei et al., 2021; Sun et al., 2022b; Ma et al., 2022; Liu et al., 2023). The propagation structure-based rumor detection models focus on capturing structural features to better detect rumors. Various propagation structure-based methods have been extensively proposed. Bian et al. (2020) model the propagation structure from both top-down and bottom-up to study the bidirectional spread of rumors. Works (He et al., 2021; Sun et al., 2022b; Liu et al., 2023; Cui and Jia, 2024) use augmentation techniques to construct augmented graphs and leverage contrastive learning to capture the propagation process. Tao et al. (2024) decompose the graph structure into subgraph components to model the bidirectional propagation process. Moreover, in addition to the aforementioned static graphs, dynamic graphs have also gained increasing attention for rumor detection. Some works (Lao et al., 2021; Chang et al., 2024) incorporate temporal information as part of the node features to capture the dynamic modeling process of the propagation graph. Works (Choi et al., 2021; Sun et al., 2022a; Xu et al., 2024) model the dynamic process of rumor propagation by dividing the dynamic graph into different graph snapshots to model the dynamics of rumor propagation. However, these methods overlook the dynamic nature of rumor spread, such as varying propagation rates, the depth of influence over time, and the breadth across different social groups. Addressing these temporal features is crucial for improving the performance of rumor detection.

2.2 Dynamic Graph Neural Networks

Graphs are used to represent relationships or connections between nodes and provide an effective means of describing and modeling complex systems and structures in the real world. Graph Neural Networks (GNNs), such as GCN(Kipf and Welling, 2016), GAT(Veličković et al., 2017), and GIN(Xu et al., 2019), combine graph-based computations with deep learning techniques, achieving outstanding performance across various graph-related tasks. To further explore the development of dynamic graphs in real-world, Dynamic GNNs (DGNNs) integrate temporal information with GNNs to capture both structural information and temporal information. The dynamic GNN models have shown exceptional performance across a wide range of tasks and attracted significant attention. GNNs and RNNs are often utilized in dynamic graphs, with GNNs employed for processing graph structures and RNNs used for handling temporal information (Liang et al., 2023; Li et al., 2024; Casadesus-Vila et al., 2024). Works (You et al., 2022; Zhu et al., 2023) focus on snapshot updates and fusion in dynamic graphs. Some study (Wang et al., 2021; He et al., 2023) underscore the integration of temporal and graph-based information to investigate the effectiveness of dynamic graph modeling. Different from the above works, we propose the adaptive sliding window and memory-augmented attention to capture the dynamics of rumors. Our study focuses on modeling rumor dynamics under real-world temporal contexts.

3 Methodology

3.1 Problem Definition

Rumor detection can be defined as a classification task. Formally, let $\mathcal{C} = \{C_1, C_2, \ldots, C_n\}$ be the rumor detection dataset, where C_i is the i-th event and n is the number of events. For each event $C = \{P_1, P_2, \ldots, P_{|C|}, G\}$, P_1 is the source post, other P_j represents the j-th responsive post, and $|C|$ is the number of posts in the event C. All posts in the event C are ordered chronologically and the set of timestamps for posts is denoted as $T = \{t_1, t_2, \ldots, t_{|C|}\}$, where $t_1 = 0$ represents the timestamp of the source post and other t_j represents the timestamp of the j-th responsive post. $G = \langle V, A, X \rangle$ is the propagation graph with the root node P_1, where V refers to the set of nodes corresponding to posts. $A \in \{0,1\}^{|C|\times|C|}$ represents the adjacency matrix, where if there is a response relationship between node P_u and P_v, $A_{u,v} = A_{v,u} = 1$, otherwise $A_{u,v} = A_{v,u} = 0$. $X \in \mathbb{R}^{|C|\times d}$ denotes the node feature matrix, where d is the node embedding dimension. Rumor detection aims to learn a function $f : \mathcal{C} \rightarrow \mathcal{Y}$ that classifies each event into one of the categories $\mathcal{Y} \in \{F, T\}$ (i.e., Rumor or Non-Rumor).

3.2 Overview

We propose a novel adaptive Sliding Window with memory-augmented Attention Model (SWAM) for rumor detection. As illustrated in Figure 2, we will explain three steps in a pipeline which include initialization, adaptive sliding window, and memory-augmented attention.

3.3 Adaptive Sliding Window

In the real world, the rate of event propagation varies across different stages. To model this phenomenon, we propose an adaptive sliding window to partition nodes of a rumor propagation graph into multiple windows. This mechanism allows for adaptive adjustment of the window size based on the dynamics of rumor spread, making it more flexible and suitable for modeling the spread of rumors over time.

Specifically, for an event C, the adaptive sliding window divide the propagation graph into k sequential windows $\mathcal{W} = \{\mathcal{W}_1, \mathcal{W}_2, \ldots, \mathcal{W}_k\}$, with each window containing a different number of nodes. $\mathcal{W}_1$ serves as the base for the subsequent windows. For subsequent windows, the adjustment of the window size in $\mathcal{W}_i$ is determined by the propagation rate of nodes in $\mathcal{W}_{i-1}$.

3.3.1 Window Initialization

For an event C, given an initial set of nodes as the first window $\mathcal{W}_1 = \{P_1, P_2, \ldots, P_{|\mathcal{W}_1|}\}$, where $\mathcal{W}_1$ contains the source post and early responsive posts, and $|\mathcal{W}_1|$ is the number of nodes in $\mathcal{W}_1$. The corresponding timestamp set is $T_1 = \{t_1, t_2, \ldots, t_{|\mathcal{W}_1|}\}$.

We introduce a propagation rate function $r(\mathcal{W})$, which calculates the propagation rate of nodes within a window based on timestamp. Formally, given a window $\mathcal{W}_i$, its timestamp set is $T_i = \{t_i, t_{i+1}, \ldots, t_{|\mathcal{W}_i|}\}$. The propagation rate is calculated as follows:

$$\Delta t_j = t_{j+1} - t_j, j \in [i, |\mathcal{W}_i| - 1] \quad (1)$$

$$r(\mathcal{W}_i) = \frac{\sum_{j=1}^{|\mathcal{W}_i'|} \frac{1}{\Delta t_j}}{|\mathcal{W}_i'|} \quad (2)$$

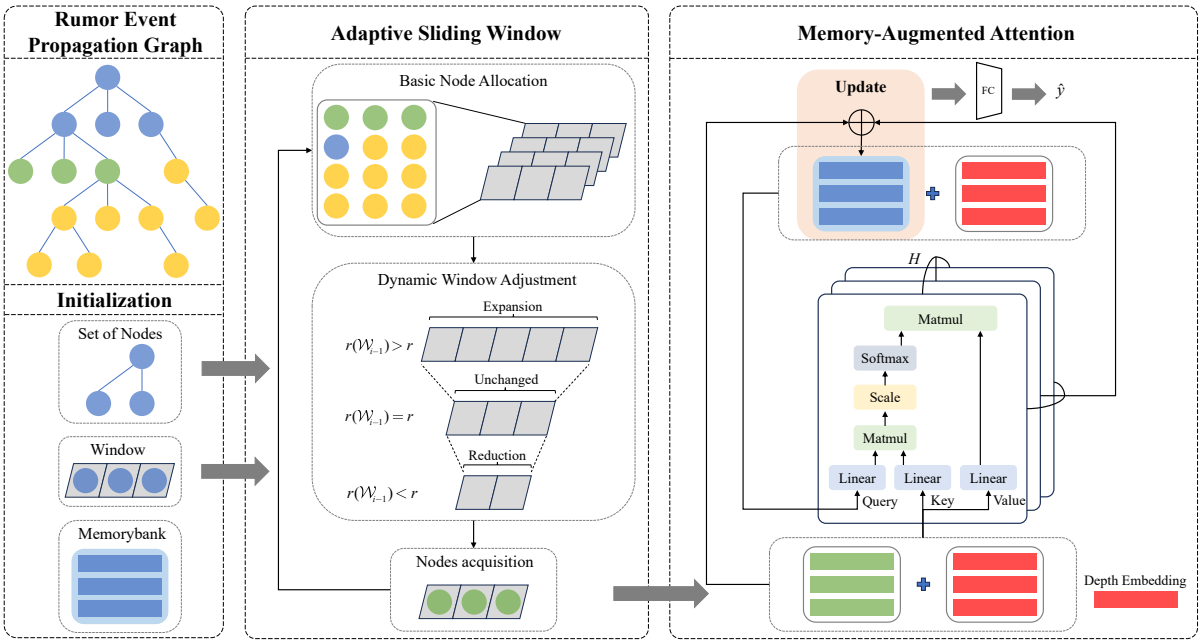


Figure 2: Overview of the proposed framework SWAM which takes three steps in a pipeline: initialization, adaptive sliding window, and memory-augmented attention.

where Δt_j represents timestamp intervals, $\frac{1}{\Delta t_j}$ represents the propagation rate per unit of time, and $|\mathcal{W}_i'|$ is the number of non-zero timestamp intervals.

We calculate the propagation rate $r(\mathcal{W}_1)$ of nodes in $\mathcal{W}_1$, defining it as the base rate r for adjusting the size of subsequent windows.

$$r = r(\mathcal{W}_1) \tag{3}$$

3.3.2 Dynamic Window Adjustment

The acquisition of subsequent window nodes is carried out in two stages: basic node allocation and dynamic window adjustment.

For the i-th window $\mathcal{W}_i$, we first evenly distribute the remaining $|C - \sum_{j=1}^{i-1} \mathcal{W}_j|$ nodes across the remaining $k - (i-1)$ windows.

$$b = \frac{|C - \sum_{j=1}^{i-1} \mathcal{W}_j|}{k - (i-1)} \tag{4}$$

where b represents the number of basic node allocations.

Then we calculate the propagation rate $r(\mathcal{W}_{i-1})$ of the window $\mathcal{W}_{i-1}$. If $r(\mathcal{W}_{i-1})$ exceeds r, we expand the size of $\mathcal{W}_i$ by adding more nodes based on b. If $r(\mathcal{W}_{i-1})$ is below r, we reduce the size of $\mathcal{W}_i$ by removing nodes based on b. If $r(\mathcal{W}_{i-1})$ equals r, the size of $\mathcal{W}_i$ remains unchanged.

$$|\mathcal{W}_i| = \begin{cases} b + \alpha b & r(\mathcal{W}_{i-1}) > r \\ b & r(\mathcal{W}_{i-1}) = r \\ b - \alpha b & r(\mathcal{W}_{i-1}) < r \end{cases} \tag{5}$$

where $|\mathcal{W}_i|$ represents the final number of nodes contained in $\mathcal{W}_i$. α is a hyperparameter utilized to control the window size.

After obtaining the nodes in $\mathcal{W}_i$, we update $r(\mathcal{W}_{i-1})$ to the new base rate.

$$r \leftarrow r(\mathcal{W}_{i-1}) \tag{6}$$

The adjustment process is iteratively applied to subsequent windows until all nodes are assigned. Finally, we can obtain the windows $\mathcal{W}$ that includes the dynamic variation in the number of nodes and ensures that each window reflects the underlying rumor spread effectively.

$$\mathcal{W} = \{\mathcal{W}_1, \mathcal{W}_2, \ldots, \mathcal{W}_k\} \tag{7}$$

3.4 Memory-Augmented Attention

To fully leverage the nodes within the window, we propose a novel attention mechanism designed to iteratively update a memorybank with incoming nodes from different windows in a structured manner. This mechanism combines the multi-head attention mechanism with node depth information,

dynamically updating the memorybank by leveraging both node features and structural characteristics. The key idea is to treat the nodes in the first window as initial memory states and gradually incorporate the nodes from subsequent windows as incremental updates.

Specifically, for the sliding window $\mathcal{W}$ of event C, the nodes in $\mathcal{W}_1$ serve as the initial memorybank $\mathcal{M}$ and the subsequent windows $\{\mathcal{W}_i\}_{i=2}^{k}$ act as incremental nodes. We update $\mathcal{M}$ with the nodes from subsequent windows through attention-based interaction.

3.4.1 Depth Embedding Layer

Since the self-attention mechanism does not inherently account for the depth of nodes in the propagation structure, we introduce a node depth embedding layer to incorporate depth-related information. This depth embedding provides a basis for encoding hierarchical relationships between nodes. This helps capture the hierarchical structure of the rumor propagation graph and the relative position of nodes within the graph.

To calculate the depths of the nodes in both $\mathcal{W}_1$ and $\mathcal{W}_2$, we first merge the nodes from the two windows into a unified propagation graph. Specifically, we construct a new adjacency matrix by combining the adjacency matrices A_1 of $\mathcal{W}_1$ and A_2 of $\mathcal{W}_2$. Then we map each node to a unique depth value, which can capture the node's level in the propagation structure and help the model better understand the structural dependencies among nodes within the graph.

$$depth = ComputeDepth([A_1; A_2]) \tag{8}$$

$$\mathcal{D} = E_{DE}(depth) \tag{9}$$

where $ComputeDepth$ is a function that calculates the depth of each node in the propagation graph. E_{DE} denotes the trainable depth embedding layer, and $\mathcal{D}$ denotes the depth embedding.

3.4.2 Memorybank Initialization

We initialize the memorybank $\mathcal{M}$ using the nodes from $\mathcal{W}_1$. Specifically, we encode the feature matrix X_1 of $\mathcal{W}_1$ using an MLP to obtain the initial state of $\mathcal{M}$.

$$\mathcal{M} = \text{MLP}(X_1) \tag{10}$$

Similarly, we also encode X_2 of $\mathcal{W}_2$ to obtain the representations of incremental nodes.

$$\mathcal{S} = \text{MLP}(X_2) \tag{11}$$

3.4.3 Incremental Updates with Multi-Head Attention

To update the memorybank efficiently, we apply multi-head attention mechanism between the memorybank $\mathcal{M}$ and incremental nodes $\mathcal{S}$. To utilize the structural position of each node in the graph, we introduce the node depth information based on node features.

$$\hat{\mathcal{M}} = \mathcal{M} + \mathcal{D}_{A_1} \tag{12}$$

$$\hat{\mathcal{S}} = \mathcal{S} + \mathcal{D}_{A_2} \tag{13}$$

Since not all incremental nodes contribute equally to the memorybank, we design memory-augmented attention to measure the importance of incremental nodes with respect to memorybank. In this attention mechanism, queries come from the representation $\hat{\mathcal{M}}$, keys and values come from $\hat{\mathcal{S}}$. By calculating the similarity between memorybank and incremental nodes, incremental nodes are assigned different weights to represent the importance.

$$Q = W_Q\hat{\mathcal{M}}, K = W_K\hat{\mathcal{S}}, V = W_V\hat{\mathcal{S}} \tag{14}$$

$$Attn(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{15}$$

$$\mathcal{Z} = \mathop{\Big\|}_{i=1}^{H} Attn_i(Q, K, V) \tag{16}$$

where Q, K and V are the queries, keys and values transformed by trainable parameter matrices W_Q, W_K and W_V. d_k is the dimension of queries and keys. $\|$ is the concatenation operation. H is the number of heads.

We concatenate the multi-head attention representation $\mathcal{Z}$ with the incremental node representation $\mathcal{S}$ to incrementally update the memorybank. The final concatenated result is updated to memorybank $\mathcal{M}$.

$$\mathcal{M} \leftarrow Concat(\mathcal{Z}, \hat{\mathcal{S}}) \tag{17}$$

3.4.4 Iterative Update Process

This process of applying multi-head attention mechanism to the memorybank and incremental nodes is repeated iteratively for subsequent window. Finally, we obtain a final memorybank $\mathcal{M}$ that incorporates all the information from the nodes across all windows. We then apply mean-pooling operators to obtain the final representation z of the event.

$$z = \text{MEAN}(\mathcal{M}) \tag{18}$$

Method	Class	TWITTER				DRWeibo			
		Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1
Bi-GCN	F	0.7591	0.7595	0.7419	0.7506	0.8118	0.8033	0.8031	0.8032
	T		0.7771	0.7732	0.7751		0.8256	0.8187	0.8221
EBGCN	F	0.7539	0.7449	0.7428	0.7438	0.8212	0.8080	0.8116	0.8098
	T		0.7657	0.7640	0.7648		0.8333	0.8303	0.8318
GACL	F	0.7609	0.7987	0.6781	0.7335	0.8541	0.8454	0.8463	0.8458
	T		0.7454	0.8377	0.7889		0.8628	0.8609	0.8618
RDEA	F	0.7855	0.7942	0.7496	0.7713	0.8610	0.8564	0.8482	0.8523
	T		0.7857	0.8181	0.8016		0.8656	0.8723	0.8689
TrustRD	F	0.7695	0.7751	0.7359	0.7550	0.8500	0.8528	0.8278	0.8401
	T		0.7749	0.7974	0.7860		0.8494	0.8698	0.8595
DynGCN	F	0.7693	0.7647	0.7495	0.7570	0.8306	0.8197	0.8252	0.8225
	T		0.7759	0.7893	0.7825		0.8427	0.8354	0.8390
PSGT	F	0.8089	0.8148	0.7814	0.7977	0.8427	0.8342	0.8336	0.8339
	T		0.8141	0.8332	0.8235		0.8529	0.8510	0.8520
SWAM	F	**0.8275**	**0.8356**	**0.7964**	**0.8156**	**0.8839**	**0.8792**	**0.8746**	**0.8769**
	T		**0.8258**	**0.8560**	**0.8406**		**0.8886**	**0.8922**	**0.8904**

Table 1: Rumor detection results on TWITTER and DRWeibo datasets. Abbrev.: Rumor (F), Non-Rumor (T).

3.5 Training Objective

To calculate the labels of the rumors, we apply a fully connected layer followed by a softmax layer,

$$\hat{y} = softmax(W_f z + b_f) \tag{19}$$

where $\hat{y}$ is the predicted probability distribution. W_f and b_f are weight and bias parameters respectively.

Our rumor detection model is trained with the cross-entropy loss $\mathcal{L}_C$ of the predictions and ground truth labels, which defined as:

$$\mathcal{L}_C = -\sum_{i}^{|\mathcal{Y}|} y_i log\hat{y}_i \tag{20}$$

where y_i denotes ground-truth label and $\hat{y}_i$ denotes the predicted probability distribution for the i-th event.

4 Experiments

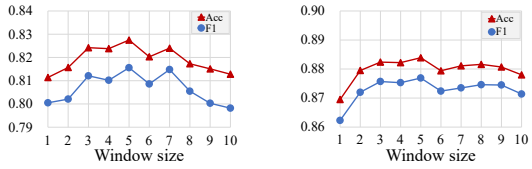
Refer to Appendix A for details of the datasets, experimental setup and comparison models.

4.1 Experimental Results

Table 1 presents the rumor detection results of various methods on the TWITTER and DRWeibo datasets. The results clearly demonstrate the strong competitiveness of SWAM across both datasets. In terms of all evaluation metrics, SWAM consistently outperforms other models. Notably, its performance surpasses that of state-of-the-art models such as PSGT and TrustRD. This confirms the advantage of combining adaptive sliding window and memory-augmented attention for learning the evolution of rumor propagation. Although BiGCN utilizes the rumor propagation structure and EBGCN employs edge enhancement to explore latent relationships within the propagation graph, neither method captures the key features of how rumors and non-rumors spread at different stages of the propagation process. Similarly, while significant performance improvements have been achieved in rumor detection by combining graph augmentation and contrastive learning in methods like GACL, RDEA, and TrustRD, they fail to account for the dynamic evolution of rumor propagation. The use of transformers in PSGT, while effective in capturing dependencies between posts, similarly overlooks the dynamic nature of rumor propagation, leading to inferior performance compared to SWAM. Compared to the static graph-based approaches mentioned above, DynGCN models the dynamic propagation of rumors. It constructs multiple temporal snapshots and sequential snapshots for rumor detection but fails to explicitly differentiate the time span of rumor events and the depth of their propagation paths. In contrast, SWAM model leverages the adaptive sliding window to deal with the dynamic and uneven nature of rumor propagation. Additionally, it employs memory-augmented attention to capture long-range dependencies between posts.

Method	Class	TWITTER		DRWeibo	
		Acc.	F1	Acc.	F1
SWAM	F	**0.8275**	**0.8156**	**0.8839**	**0.8769**
	T		**0.8406**		**0.8904**
w/o D	F	0.8190	0.8066	0.8607	0.8520
	T		0.8323		0.8686
w/o S	F	0.8114	0.8005	0.8695	0.8623
	T		0.8252		0.8770
w/o A	F	0.7908	0.7727	0.8584	0.8494
	T		0.8104		0.8670

Table 2: Results of ablation study.



(a) TWITTER dataset (b) DRWeibo dataset

Figure 3: Performance of sensitivity of window size.

4.2 Ablation Study

We demonstrate the effectiveness of the SWAM framework from two aspects: the use of the adaptive sliding window and the incorporation of memory-augmented attention. The variants are defined as follows: (1) w/o D: removing the node depth information; (2) w/o S: removing the adaptive sliding window; (3) w/o A: removing the memory-augmented attention.

As shown in Table 2, when we remove the depth information from the memory-augmented attention, the accuracy on TWITTER and DRWeibo drops by 0.85% and 2.32%. Depth information reflects a node's hierarchical position or level within the propagation structure. By incorporating depth information into the node features, the model can better capture the hierarchical relationships in the graph. When we remove the adaptive sliding window, the performance on TWITTER and DRWeibo significantly decreases by 1.61% and 1.44%. The adaptive sliding window adjusts the number of nodes according to the propagation rate, providing a dynamic reflection of the rumor's spread intensity and trends over time. This approach allows for more accurate identification of key moments and important nodes in the rumor propagation process. When the entire attention mechanism is removed, the model's performance deteriorates significantly. We can observe that the model can flexibly handle varying numbers of incremental nodes with the attention mechanism. By continuously updating the memorybank, the model accumulates historical information and makes node feature representations not only depend on neighboring nodes but also consider the overall information.

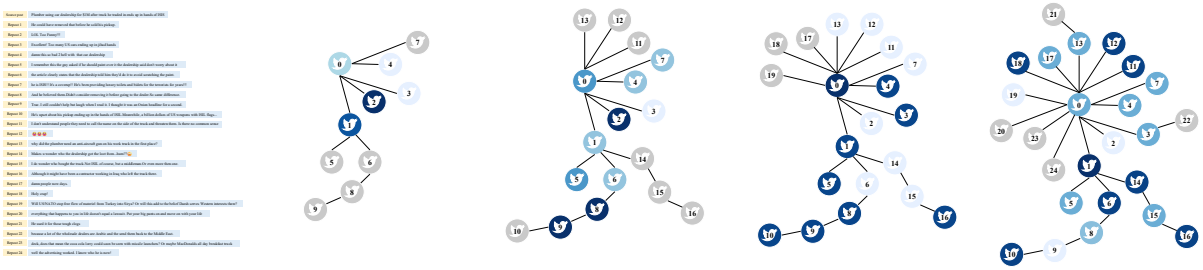
4.3 Sensitivity of Window Size

In the SWAM framework, the number of sliding windows k is a crucial parameter. To assess the impact of this hyperparameter on model performance, we experiment with different window size to observe the performance variations.

As shown in Figure 3, when the number of windows is small, the model's performance on both datasets is suboptimal. A limited number of windows may compress the propagation process, which causes some windows to contain an excessive number of nodes. The slower-spreading nodes may not be sufficiently represented and make it difficult to capture finer-grained propagation paths accurately. As the number of windows increases beyond a certain threshold, the model can dynamically evolve to capture more nuanced propagation changes. It can reveal hierarchical structures and variations in propagation rates during the rumor spread and ultimately achieve optimal results. As the number of windows continues to increase, although the propagation tree can be divided more finely, the number of nodes in each window decreases. This results in each window covering a more localized segment of the propagation process and reduce the ability to capture long-range propagation patterns. It can be observed that TWITTER exhibits significant fluctuations in performance during the rising phase (windows 3-7), followed by a sharp decline in later stages. This is due to the relatively short propagation time on TWITTER, where information spreads quickly between nodes, making stability more susceptible to fluctuations. During the ascending phase (spanning 3-5 windows), the curve of DRWeibo remains relatively stable. Although performance declines slightly as the number of windows increases, the propagation process is slow and sustained over a longer duration and allow it to maintain a relatively stable state.

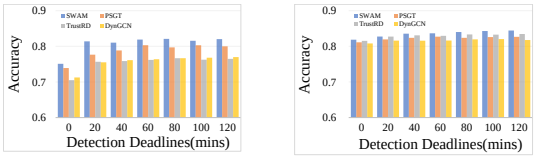
4.4 Early Rumor Detection

This experiment investigates the task of early-stage rumor detection on social media, aiming to enable timely identification of rumors. Following the method in Sun et al. (2022b), we formulate the task by defining a series of detection deadlines.



(a) The sequence of posts(b) The 1st incre. updates(c) The 2nd incre. updates(d) The 3rd incre. updates(e) The 4th incre. updates

Figure 4: A memory-augmented attention example from TWITTER. We use **blue** to represent the attention values of the nodes in memorybank, with the darker the color, the greater the attention value. We use **gray** to represent the incremental nodes. The numbers corresponding to the nodes represent the posting order. **incre.** is an abbreviation for "incremental".



(a) TWITTER dataset (b) DRWeibo dataset

Figure 5: Results of early rumor detection.

Figure 5 presents the early detection performance of SWAM, compared with PSGT, TrustRD, and DynGCN across different deadlines. As the detection deadline increases, the accuracy of all models improves. Notably, SWAM consistently outperforms the baselines at each deadline, highlighting its effectiveness in early rumor detection.

4.5 Case Study

The memory-augmented attention in SWAM is capable of capturing the key nodes that significantly influence rumor propagation at various stages. To visually demonstrate this effect, we conducted a case study on the attention distribution during rumor propagation. We select an event from the TWITTER dataset for illustration. The content of source post is "Plumber suing car dealership for $1M after truck he traded in ends up in the hands of ISIS", which is a rumor event. The rumor event has been reposted 24 times over time. Figure 4a shows the content of the source post and the reposts, and the four incremental updates of the memorybank are shown from Figure 4b to Figure 4e.

During the first incremental update, it is observed that node 2 has the darkest color. The incremental nodes contain factual descriptions as well as sarcasm. As a result, the model shows varying levels of attention to nodes of different types in the memorybank. Specifically, the sarcastic tone raises doubts about whether the event is a rumor, triggering the feature keywords related to rumor detection, which leads the model to focus more on these types of nodes. The situation is similar for the second incremental update. The nodes in the third incremental update exhibit a clear bias towards criticism, leading to greater attention being given to nodes with similar expressions in the memorybank. In the fourth incremental update, different nodes receive varying levels of attention, revealing the key nodes in the spread of rumors. Overall, the memory-augmented attention can accurately capture the long-term dependencies between rumors and identify the key features of their propagation.

5 Conclusion

In this paper, we propose a novel adaptive Sliding Window with memory-augmented Attention Model (SWAM) for rumor detection. SWAM takes three steps in a pipeline: initialization, adaptive sliding window, and memory-augmented attention. The initial window consists of the source post and early responsive posts. We propose an adaptive sliding window to partition nodes of a rumor propagation graph into multiple disjoint windows. A novel memory-augmented attention is designed to iteratively update a memorybank with incoming nodes from different windows. This mechanism combines the multi-head attention mechanism with node depth information. Experiments on two public datasets demonstrate that SWAM outperforms the state-of-the-art baselines. In the future, we will explore more approaches for temporal modeling to enhance the performance of rumor detection further.

Limitations

One limitation of our model is that the constructed temporal information does not account for multi-scale temporal encoding. If the dynamic changes of an event are associated with different time scales (such as minutes, hours, or days), it may lead to suboptimal performance. In the future, we will explore more approaches for temporal modeling to enhance the performance of rumor detection further.

References

Tian Bian, Xi Xiao, Tingyang Xu, Peilin Zhao, Wenbing Huang, Yu Rong, and Junzhou Huang. 2020. Rumor detection on social media with bi-directional graph convolutional networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 549–556.

Guillem Casadesus-Vila, Joan-Adria Ruiz-de Azua, and Eduard Alarcon. 2024. Toward autonomous cooperation in heterogeneous nanosatellite constellations using dynamic graph neural networks. *arXiv preprint arXiv:2403.00692*.

Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. 2011. Information credibility on twitter. In *Proceedings of the 20th international conference on World wide web*, pages 675–684.

Ya-Ting Chang, Zhibo Hu, Xiaoyu Li, Shuiqiao Yang, Jiaojiao Jiang, and Nan Sun. 2024. Dihan: A novel dynamic hierarchical graph attention network for fake news detection. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 197–206.

Jiho Choi, Taewook Ko, Younhyuk Choi, Hyungho Byun, and Chong-kwon Kim. 2021. Dynamic graph convolutional networks with attention mechanism for rumor detection on social media. *Plos one*, 16(8):e0256039.

Chaoqun Cui and Caiyan Jia. 2024. Propagation tree is not deep: Adaptive graph contrastive learning approach for rumor detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 73–81.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Yingtong Dou, Kai Shu, Congying Xia, Philip S Yu, and Lichao Sun. 2021. User preference-aware fake news detection. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*, pages 2051–2055.

Yaqian Dun, Kefei Tu, Chen Chen, Chunyan Hou, and Xiaojie Yuan. 2021. Kan: Knowledge-aware attention network for fake news detection. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 81–89.

Song Feng, Ritwik Banerjee, and Yejin Choi. 2012. Syntactic stylometry for deception detection. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 171–175.

Bowei He, Xu He, Yingxue Zhang, Ruiming Tang, and Chen Ma. 2023. Dynamically expandable graph convolution for streaming recommendation. In *Proceedings of the ACM Web Conference 2023*, pages 1457–1467.

Zhenyu He, Ce Li, Fan Zhou, and Yi Yang. 2021. Rumor detection on social media with event augmentations. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*, pages 2020–2024.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Sejeong Kwon, Meeyoung Cha, Kyomin Jung, Wei Chen, and Yajun Wang. 2013. Prominent features of rumor propagation in online social media. In *2013 IEEE 13th international conference on data mining*, pages 1103–1108. IEEE.

An Lao, Chongyang Shi, and Yayi Yang. 2021. Rumor detection with field of linear and non-linear propagation. In *Proceedings of the Web Conference 2021*, pages 3178–3187.

Hongxi Li, Zuxuan Zhang, Dengzhe Liang, and Yuncheng Jiang. 2024. K-truss based temporal graph convolutional network for dynamic graphs. In *Asian Conference on Machine Learning*, pages 739–754. PMLR.

Ke Liang, Lingyuan Meng, Meng Liu, Yue Liu, Wenxuan Tu, Siwei Wang, Sihang Zhou, and Xinwang Liu. 2023. Learn from relational correlations and periodic events for temporal knowledge graph reasoning. In *Proceedings of the 46th international ACM SIGIR conference on research and development in information retrieval*, pages 1559–1568.

Hongzhan Lin, Jing Ma, Liangliang Chen, Zhiwei Yang, Mingfei Cheng, and Chen Guang. 2022. Detect rumors in microblog posts for low-resource domains via adversarial contrastive learning. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 2543–2556.

Leyuan Liu, Junyi Chen, Zhangtao Cheng, Wenxin Tai, and Fan Zhou. 2023. Towards trustworthy rumor detection with interpretable graph structural learning. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 4089–4093.

Yang Liu and Yi-Fang Wu. 2018. Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Guanghui Ma, Chunming Hu, Ling Ge, Junfan Chen, Hong Zhang, and Richong Zhang. 2022. Towards robust false information detection on social networks with contrastive learning. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 1441–1450.

Jiachen Ma, Jing Dai, Yong Liu, Meng Han, and Chunyu Ai. 2023. Contrastive learning for rumor detection via fitting beta mixture model. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 4160–4164.

Jing Ma, Wei Gao, Prasenjit Mitra, Sejeong Kwon, Bernard J Jansen, Kam-Fai Wong, and Meeyoung Cha. 2016. Detecting rumors from microblogs with recurrent neural networks.

Jing Ma, Wei Gao, and Kam-Fai Wong. 2018. Rumor detection on twitter with tree-structured recursive neural networks. Association for Computational Linguistics.

Jing Ma, Wei Gao, and Kam-Fai Wong. 2019. Detect rumors on twitter by promoting information campaigns with generative adversarial learning. In *The world wide web conference*, pages 3049–3055.

Erxue Min, Yu Rong, Yatao Bian, Tingyang Xu, Peilin Zhao, Junzhou Huang, and Sophia Ananiadou. 2022. Divide-and-conquer: Post-user interaction network for fake news detection on social media. In *Proceedings of the ACM web conference 2022*, pages 1148–1158.

Van-Hoang Nguyen, Kazunari Sugiyama, Preslav Nakov, and Min-Yen Kan. 2020. Fang: Leveraging social context for fake news detection using graph representation. In *Proceedings of the 29th ACM international conference on information & knowledge management*, pages 1165–1174.

Mengzhu Sun, Xi Zhang, Jiaqi Zheng, and Guixiang Ma. 2022a. Ddgcn: Dual dynamic graph convolutional networks for rumor detection on social media. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pages 4611–4619.

Tiening Sun, Zhong Qian, Sujun Dong, Peifeng Li, and Qiaoming Zhu. 2022b. Rumor detection on social media with graph adversarial contrastive learning. In *Proceedings of the ACM Web Conference 2022*, pages 2789–2797.

Xiang Tao, Liang Wang, Qiang Liu, Shu Wu, and Liang Wang. 2024. Semantic evolvement enhanced graph autoencoder for rumor detection. In *Proceedings of the ACM on Web Conference 2024*, pages 4150–4159.

Lin Tian, Xiuzhen Jenny Zhang, and Jey Han Lau. 2022. Duck: Rumour detection on social media by modelling user and comment propagation networks. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4939–4949.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Yanbang Wang, Pan Li, Chongyang Bai, and Jure Leskovec. 2021. Tedic: Neural modeling of behavioral patterns in dynamic social interaction networks. In *Proceedings of the Web Conference 2021*, pages 693–705.

Lingwei Wei, Dou Hu, Wei Zhou, Zhaojuan Yue, and Songlin Hu. 2021. Towards propagation uncertainty: Edge-enhanced Bayesian graph convolutional networks for rumor detection. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3845–3854, Online. Association for Computational Linguistics.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How powerful are graph neural networks? In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Shuai Xu, Jianqiu Xu, Shuo Yu, and Bohan Li. 2024. Identifying disinformation from online social media via dynamic modeling across propagation stages. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 2712–2721.

Weizhi Xu, Junfei Wu, Qiang Liu, Shu Wu, and Liang Wang. 2022. Evidence-aware fake news detection with graph neural networks. In *Proceedings of the ACM web conference 2022*, pages 2501–2510.

Fan Yang, Yang Liu, Xiaohui Yu, and Min Yang. 2012. Automatic detection of rumor on sina weibo. In *Proceedings of the ACM SIGKDD workshop on mining data semantics*, pages 1–7.

Jiaxuan You, Tianyu Du, and Jure Leskovec. 2022. Roland: graph learning framework for dynamic graphs. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pages 2358–2366.

Junyou Zhu, Chao Gao, Ze Yin, Xianghua Li, and Jürgen Kurths. 2024. Propagation structure-aware graph transformer for robust and interpretable fake news detection. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4652–4663.

Yifan Zhu, Fangpeng Cong, Dan Zhang, Wenwen Gong, Qika Lin, Wenzheng Feng, Yuxiao Dong, and Jie Tang. 2023. Wingnn: Dynamic graph neural networks with random gradient aggregation window. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3650–3662.

Statistics	TWITTER	DRWeibo
# Events	1077	5998
# Posts	60207	376659
# Non-Rumors	564	3160
# Rumors	513	2838
Avg. time length/tree	416 Hours	1946 Hours
Avg. depth/tree	4.55	2.15

Table 3: Statistics of the datasets.

A Appendix

A.1 Datasets

We evaluate the proposed model on two real-world datasets: TWITTER (Lin et al., 2022) and DRWeibo (Cui and Jia, 2024). TWITTER is an English dataset from Twitter and DRWeibo is a Chinese dataset from Weibo. Both datasets contain the post content, propagation structure and temporal information. There are two binary labels in the datasets: Rumor (F) and Non-Rumor (T). The statistics of datasets are shown in Table 3. For the TWITTER and DRWeibo datasets, we follow (Sun et al., 2022b; Ma et al., 2023), and join the source post with each responsive post in a *[CLS] Source Post [SEP] Responsive Post [SEP]* manner to emphasize the importance of the source post.

A.2 Implementation Details

The proposed model is implemented in PyTorch framework. The parameters are optimized using Adam algorithm. The learning rate is set to 5e-4 and the batch size is set to 128. BERT (Devlin et al., 2018) is used to encode the post content. The node embedding dimension d is 768. α is 0.4. k is set to 5. The number of attention heads H is 8. The dropout of multi-head attention mechanism is 0.1. The node hidden dimension is set to 128. We adopt the evaluation method from (Bian et al., 2020) and perform 10 runs of 5-fold cross-validation to report the final results. The Accuracy (Acc.), Precision (Prec.), Recall (Rec.), and F1-score (F1) are used to measure the models.

A.3 Comparison Models

We compare the proposed model with the following baselines:

- **Bi-GCN** (Bian et al., 2020) employs a top-down and bottom-up propagation framework to capture the patterns of rumor propagation and dispersion.

- **EBGCN** (Wei et al., 2021) explores the edge-enhanced rumor detection to capture propagation features.
- **GACL** (Sun et al., 2022b) is a rumor detection model using adversarial and contrastive learning.
- **RDEA** (He et al., 2021) incorporates augmentation strategies and self-supervised learning for detection.
- **TrustRD** (Liu et al., 2023) builds self-supervised learning and Bayesian network to identify rumors.
- **DynGCN** (Choi et al., 2021) focuses on the dynamic graph to model graph snapshots and attention mechanisms representations.
- **PSGT** (Zhu et al., 2024) develops a propagation structure-aware transformer to obtain propagation relationships for interpretable rumor detection.