Fake News Detection Through Temporally Evolving User Interactions

Shuzhi Gong[1], Richard O. Sinnott[1], Jianzhong Qi[1], and Cecile Paris[2]

[1] The University of Melbourne, Melbourne VIC 3010, Australia
{shuzhi, rsinnott, jianzhong.qi}@unimelb.edu.au
[2] Data61 CSIRO, Sydney NSW 1710, Australia
Cecile.Paris@data61.csiro.au

Abstract. Detecting fake news on social media is an increasingly important problem, because of the rapid dissemination and detrimental impact of fake news. Graph-based methods that encode news propagation paths into tree structures have been shown to be effective. Existing studies based on such methods represent the propagation of news through static graphs or coarse-grained graph snapshots. They do not capture the full dynamics of graph evolution and hence the temporal news propagation patterns. To address this issue and model dynamic news propagation at a finer-grained level, we propose a temporal graph-based model. We join this model with a neural Hawkes process model to exploit the distinctive self-exciting patterns of true news and fake news on social media. This creates a highly effective fake news detection model that we named SEAGEN. Experimental results on real datasets show that SEAGEN achieves an accuracy of fake news detection of over 93% with an advantage of over 2.5% compared to other state-of-the-art models.

Keywords: Fake News Detection · Dynamic Graph Embedding.

1 Introduction

Fake news created with malicious intent can lead to a substantially negative impact on society, especially during major events such as the U.S. presidential election and the COVID-19 pandemic. To combat the negative impact, various methods have been proposed including exploiting the news content [4], the characteristics of the users involved [17] and the message propagation patterns [1]. In this paper, we focus on detecting fake news propagated on social media platforms such as Twitter through its dissemination and user interactions patterns.

News propagation on social media can be represented by graph-based models where the social media posts (or users) are represented as nodes, while replies, retweets, or other dissemination actions are represented as edges. Many existing graph-based fake news detection models [1, 19] use a static graph that shows the complete spatial propagation network after a (fake) news item has been spread. However, these spatial structure-based approaches have largely oversimplified the temporal structure associated with the message propagation, i.e., the

sequence and interval of the messages propagated along the timeline. For example, in Fig. 1, three news propagation graphs (each node represents a post) share the same tree structure, but have different temporal patterns. The propagation graphs in Figs. 1a and 1b differ in the time order when the nodes v_2, v_3, and v_4 are added to the graphs. For the propagation graphs in Figs. 1a and 1c, whilst their time orders are the same, nodes v_1 to v_4 in Fig. 1c are much closer in time.

Further, it has been observed that the fake news propagation process exhibits a viral nature and has different stages in terms of people's attention and reactions, resulting in a unique life cycle [16]. In particular, fake news tend to exhibit a sudden increase in the propagation process, while true news have a much smoother process. A sudden increase can be related to the *self-exciting phenomenon* [13] caused by social bot promotions or people's rapid actions to question or correct false information [16].

Such observations motivate us to model the temporal evolving nature of user interactions (i.e., news propagation process) as the basis for detecting fake news.

Sequence-based methods (e.g., [8]) flatten the propagation graph into a chronological sequence of events. Models such as Recurrent Neural Networks (RNN) and BERT [3] can then be applied to learn temporal patterns. A limitation of such methods is that they largely overlook the graph structure of the news propagation patterns.

Other studies [2] use propagation graph snapshots to model both the spatial and temporal propagation patterns. This method only captures the graphs at selected time points, hence they may miss the exact time when a drastic change in the propagation graph occurs. Besides, these studies ignore the self-exciting phenomenon associated with fake news.

To better model the temporal propagation patterns for fake news detection, we propose the **S**elf-**E**xciting-**A**ware **G**raph **E**volution **N**etwork (`SEAGEN`) model, based on the temporal interactions associated with news propagation processes. We represent news propagation on social media using temporal graphs, where social media posts are nodes and user interactions are the edges. Different from existing graph-based methods, we encode the graph evolution process by in-

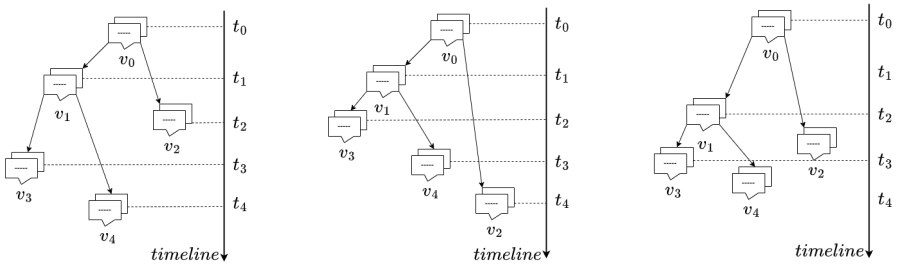

Fig. 1: Example of three news propagation graphs.

tegrating local sub-graph modeling and global evolution modeling. Our model consists of three main components: (1) a *local sub-graph encoding module*, (2) a *self-attention-based global temporal evolution module*, and (3) a *neural Hawkes process-based self-exciting module*.

The local sub-graph encoding module encodes a local sub-graph, which models the interaction between a graph node and its neighbouring nodes. Each local sub-graph represents a single temporal stage in the graph evolution process. By encoding the full sequence of local sub-graphs, `SEAGEN` learns fine-grained news propagation patterns that can then be used for fake news detection.

A Transformer-based global temporal evolution module then integrates the sequence of local sub-graphs and captures the overall temporal evolution (i.e., news propagation) process. It uses a self-attention mechanism to re-weight the (encoded) local sub-graphs based on their content and timestamp of interactions.

A neural Hawkes process-based self-exciting module models the self-exciting phenomenon using a neural network and Hawkes process. We establish fake news detection and Hawkes intensity prediction to capture the self-exciting nature of social media-based fake news.

To summarise, this paper makes the following contributions:

- we propose a novel model named `SEAGEN` for fake news detection based on a sub-graph sequence-based approach to model temporal news propagation patterns in social networks;
- to learn local propagation patterns, we propose a time-aware encoder to encode local sub-graphs that model fine-grained user interaction events;
- to learn the overall propagation patterns, we use a self-attention-based global temporal evolution module and a neural Hawkes process module, where the former module integrates patterns learned from the local sub-graphs and the latter captures the self-exciting phenomenon;
- we perform extensive experiments on social media datasets. The results show that `SEAGEN` outperforms state-of-the-art models with an overall accuracy exceeding 93% for fake news detection, including a detection accuracy increase in early stage.

2 Problem Formulation and Data Structure

We consider a fake news detection dataset from social media, which consists of a set of claims $\mathcal{C} = \{C_1, C_2, \ldots, C_{|\mathcal{C}|}\}$. Each claim $C_i = \{v_0^i, v_1^i, v_2^i, \ldots, v_{n_i}^i, G_i\}$ corresponds to a news item, where v_0^i is a source post (e.g., a source tweet). There should be n_i ($n_i \geq 0$) responding posts (e.g., retweets or replies) $\{v_1^i, v_2^i, \ldots, v_{n_i}^i\}$ listed in chronological order.

Graph $G_i = \langle V_i, E_i, T_i \rangle$ is a temporal graph (a tree) representing the propagation pattern of the posts, where v_0^i is the root node (cf. the top left graph in Fig. 2). The set of nodes $V_i = \{v_0^i, v_1^i, v_2^i, \ldots, v_{n_i}^i\}$ represents the source and the responding posts. The set of edges $E_i = \{e_{st}^i | s, t = 0, \ldots, n_i\}$ represents the responding relationships between the posts, where s and t represent the subscripts of a post and a response to it, respectively. When $s = 0$, it refers to the source

post v_0^i. The set $T_i = \{t_{st}^i | s, t = 0, \ldots, n_i\}$ represents the occurrence times of the edges in E_i. In this graph representation, each node $v_j^i \in V_i$ is represented by a vector $\mathbf{x}_j^i$, which is a text embedding of the post content. We use text embeddings from a pre-trained BERT-base text encoder [3] for simplicity.

Given graph G_i, our goal is to classify if it represents a fake or a true news.

We note that replies and retweets represent two different interactions on Twitter. Replies are comments with textual contents, while retweets re-post source posts and usually express a supportive attitude. To take advantage of the textual content of replies and the supportive information of retweets, we represent the replies with their own textual contents and retweets with the source post's textual contents, respectively.

3 Proposed Model

As shown in Fig. 2, `SEAGEN` consists of three modules: a *local sub-graph encoding* (LSGE) module, a *Transformer-based global evolution capturing* (GEC) module, and a *neural Hawkes process* module. The LSGE module extracts a sequence of sub-graphs from G_i and learns a representation for each sub-graph encoding both the structural and the temporal information (Section 3.1). The learned embeddings of the sequence of sub-graphs are fed into the GEC module that re-weights the embeddings and prepares for the neural Hawkes process (Section 3.2). The weighted embedding sequence is given to a neural Hawkes process to establish the interaction intensities (Section 3.3) and to a feedforward neural network (FFN) to determine whether G_i represents a fake news item. The output of the neural Hawkes process model and the FFN are used to compute the loss function for model training (Section 3.4).

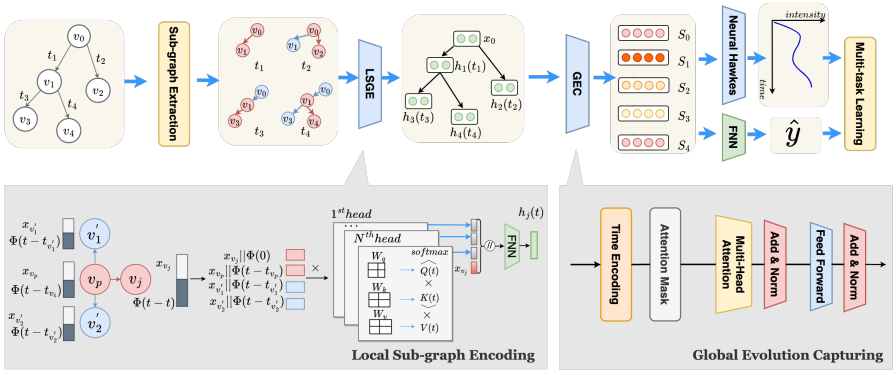

Fig. 2: Architecture of `SEAGEN`.

3.1 Local Sub-graph Encoding Module

Given graph G_i, our LSGE module first extracts a sequence of local sub-graphs. At each time t_j that corresponds to the occurrence of node $v_j^i \in G_i$ (i.e., post w_j^i) except for the source post (i.e., $j > 0$), we extract a sub-graph of G_i. The sub-graph is formed by v_j^i, v_j^i's parent node v_p^i, and v_p^i's neighbor nodes $\mathcal{N}(v_i^i; t_j)$ before time t_j. For example, as shown in Fig. 2, as time t_3, node v_3 occurs. A local sub-graph is formed by v_3, v_1 (i.e., v_3's parent node), and v_0 (i.e., a neighbour of v_1 that occurs before t_3.), in which the latter occurring node v_4 is not included. For simplification, in what follows, we omit the superscript i for the nodes in graph G_i, and we simply use t instead of t_j to represent the occurrence time of v_j.

Each local sub-graph forms a small conversation represented by the topic of discussion when v_j occurs. Then, the sub-graph is embedded by multi-head attention to form a new embedding $h_j(t)$ for node v_j, as detailed below.

When the interaction between parent node v_p and child node v_j occurs at time t, we consider v_p's neighbourhood $\mathcal{N}(v_p; t) = \{v_1^{'}, ..., v_N^{'}\}$ which takes place prior to time t. The node features of v_j, v_p, and v_p's neighbours are input together into the sub-graph encoder to produce a sub-graph representation $\mathbf{h}_j(t)$. For any node v_k, we use the text representation $\mathbf{x}_k \in \mathbb{R}^{d_0}$ to denote its node features, where d_0 is the dimension of the text representation. Every interaction between node v_k and its parent node is also associated with a posting time t_k. To compute the sub-graph representation $\mathbf{h}_j(t)$, a multi-head attention (MHA) mechanism [20] is utilised to integrate features from two interacting nodes and their neighbours. The query $\mathbf{Q}(t)$ is defined from the child node v_j which initiates the interaction. The key $\mathbf{K}(t)$ and the value $\mathbf{V}(t)$ are defined from the parent node v_p and its neighbours. Formally,

$$\mathbf{h}_j(t) = \mathbf{FFN}(\mathbf{x}_j || \hat{\mathbf{h}}(t)) \tag{1}$$

$$\hat{\mathbf{h}}(t) = \mathbf{MHA}(\mathbf{Q}(t), \mathbf{K}(t), \mathbf{V}(t)) \tag{2}$$

$$\mathbf{Q}(t) = \mathbf{x}_j || \Phi(0) \tag{3}$$

$$\mathbf{K}(t) = \mathbf{V}(t) = [\mathbf{x_p} || \Phi(t - t_p), \mathbf{x_1^{'}} || \Phi(t - t_1^{'}), ..., \mathbf{x_N^{'}} || \Phi(t - t_N^{'})] \tag{4}$$

Here, $\mathbf{h}_j(t)$ is the sub-graph representation of the current interaction and also the computed hidden representation for node v_j, $\Phi(\cdot)$ represents the generic time encoding [21], $||$ is the concatenation operator, and t_p, $\{t_1^{'}, ..., t_N^{'}\}$ are the posting times of node v_p and its neighbours $\mathcal{N}(v_p; t)$. FFN is a feedforward network and MHA is a multi-headed attention layer. FNN and MHA contain all trainable parameters of the LSGE.

After LSGE, the representation of the full temporal graph G_i consists the embeddings of the source node and the sub-graphs (each corresponding to a responsive node in G_i): $\mathbf{H} = \{\mathbf{x}_0, \mathbf{h}_1(t_1), ..., \mathbf{h}_k(t_k), ..., \mathbf{h}_N(t_N)\}$.

3.2 Global Evolution Capturing Module

Next, we employ a Transformer [20] encoder based self-attention module, Global Evolution Capturing (GEC) module, to capture the global evolution process. The output of LSGE, which is the sub-graph enhanced representation of all nodes in G_i, is fed into GEC in chronological order.

The original Transformer only considers element-wise relative positions via Positional Encoding [20]. In our case, we need to be aware the interaction absolute occurrence times for later neural Hawkes process module. Therefore, an adaptive temporal-aware Positional Encoding is formulated as:

$$[\mathbf{z}(t_j)]_k = \begin{cases} \cos(t_j/10000^{\frac{k-1}{M}}), \text{if } k \text{ is odd} \\ \sin(t_j/10000^{\frac{k}{M}}), \text{if } k \text{ is even} \end{cases} \tag{5}$$

where $\mathbf{z}(t_j) \in \mathbb{R}^M$ is the time encoding of the j-th element in the sequence, and M is the dimension of the encoding. Then the output of LSGE, i.e., $\mathbf{H}$, is first enhanced with the time encoding as follows:

$$\mathbf{H}' = \mathbf{H} + \mathbf{Z} \tag{6}$$

After adding the time encoding, $\mathbf{H}'$ is passed to a self-attention module:

$$\mathbf{S} = Softmax(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{M_K}})\mathbf{V} \tag{7}$$

$$\mathbf{Q} = \mathbf{H}'\mathbf{W}^{\mathbf{Q}}, \mathbf{K} = \mathbf{H}'\mathbf{W}^{\mathbf{K}}, \mathbf{V} = \mathbf{H}'\mathbf{W}^{\mathbf{V}}. \tag{8}$$

Here, $\mathbf{Q}$, $\mathbf{K}$, ad $\mathbf{V}$ are the query, key, and value of the self-attention [20] transformed from $\mathbf{H}'$, while $\mathbf{W}^{\mathbf{Q}}$, $\mathbf{W}^{\mathbf{K}}$, and $\mathbf{W}^{\mathbf{V}}$ are the weights of the linear transformations. Note that multi-head self-attention is also implemented.

The output $\mathbf{S}$ will be used in the neural Hawkes process module to compute the continuous conditional intensity which describe the dynamics of the news propagation process.To prevent leftward information flow (i.e., the neural Hawkes process predicts intensities by inferring future events' timestamps), attention mask is implemented like [20]. The veracity prediction is also computed based on $\mathbf{S}$, as defined by the following equation.

$$\hat{y} = \mathbf{FFN}(MeanPooling(\mathbf{S})) \tag{9}$$

3.3 Neural Hawkes Process Module

A Hawkes process [5] is a self-exciting point process. It can simulate news propagation by modelling the generation of social media posts (tweets) over a continuous time domain. The frequency of posts and responses/retweets generated is determined by an underlying intensity function which considers the impact of past posts. The intensity function $\lambda(t)$ models the self-exciting nature by summation of the impact of past posts, which is defined as:

$$\lambda(t) = \mu_0(t) + \sum_{t_k<t}^{t} \phi(t - t_k) \tag{10}$$

where $\mu_0(t)$ is the base intensity, ϕ is a so-called kernel function that is used to modulate the effect of previous events k on the intensity $\lambda(t)$.

The traditional Hawkes process oversimplifies the dynamics of point processes, and assumes that all previous events have positive impact on the occurrence of the current event. However, a user's behaviour can contribute to the spread of rumors on social media and/or curb them. To overcome this limitation, we adopt a neural Hawkes based process [22] to model the complicated self-exciting phenomenon.

The output of GEC $\mathbf{S} = \{\mathbf{s}_1, ..., \mathbf{s}_N\}$ is fed forward into the neural Hawkes process module, to compute the continuous user interaction intensity. Given a self-attentive sequence of interactions with timestamps before time t: $\mathcal{H} = \{(\mathbf{s_k}, t_k) : t_k < t\}$, the continuous intensity at time t, $\lambda(t|\mathcal{H}_t)$, is computed by:

$$\lambda(t|\mathcal{H}_t) = f(\alpha\frac{t - t_j}{t_j} + w^\top \mathbf{s_j} + b) \tag{11}$$

$$f(x) = \beta \log(1 + \exp(x/\beta)) \tag{12}$$

where α, β, and w are learnable parameters, $\mathbf{s}_j$ is the self-attentive hidden state for corresponding responsive post that occurs just before time t.

3.4 Model Training

For a sequence $\mathbf{S}$ over observation interval $[t_1, t_L]$ with a continuous conditional intensity function given as $\lambda(t|\mathcal{H}_t)$, the log-likelihood can be computed as:

$$\mathcal{L}_S = \sum_{j=1}^{L} \log \lambda(t|\mathcal{H}_t) - \int_{t_1}^{t_L} \lambda(t|\mathcal{H}_t)dt \tag{13}$$

where the left part is the event log-likelihood and the right part is the non-event log-likelihood. To calculate the integral non-event log-likelihood, Monte Carlo integration [14] is utilised.

Meanwhile, the veracity prediction loss is computed as a cross-entropy loss:

$$\mathcal{L}_C = -y \log(\hat{y_1}) - (1 - y) \log(\hat{y_0}) \tag{14}$$

where the $\hat{y} = [\hat{y_0}, \hat{y_1}]$ denotes the probability of a given piece of news to be true or false. The final loss function is the sum of L_S and L_C as weighted by γ.

$$\mathcal{L} = \mathcal{L}_C + \gamma \cdot \mathcal{L}_S \tag{15}$$

Table 1: Statistics of the Datasets.

Statistic	#source tweets	#users	#fake news	#true news	Avg. time length
`Twitter`	1,147	29,858	578	569	158 hours
`FakeNewsNet`	4,168	45,109	2,079	2,089	1,951 hours

4 Experiment

4.1 Datasets

We use public Twitter datasets `Twitter` [9] and `FakeNewsNet` [16]. The former in particular is formed by true-rumours and false-rumours from the `Twitter15` [9] and `Twitter16` [9] datasets, named as true news and fake news, respectively.

`FakeNewsNet` consists of two classes of data: true news and fake news. Table 1 summarises the datasets.

4.2 Baseline Methods

We compare with state-of-the-art fake news detection models including:

- **RvNN** [10] uses a recurrent neural network to learn discriminative features from post contents by following their non-sequential propagation structure.
- **Sta-PLAN** [7] uses a self-attention mechanism and position encoding [20] to extract textual features for sequence embedding learning.
- **STS-NN** [6] jointly models the spatial and the temporal structures of the message propagation process using a gated recurrent unit (GRU).
- **Bi-GCN** [1] represents social media posts as nodes in a graph and utilises a graph convolutional network (GCN)-based model to encode the graph.
- **Dy-GCN** [2] takes snapshots of the message evolution process, builds a graph for every snapshot, and then encodes the graph snapshots by a GCN.
- **GACL** [19] enhances Bi-GCN by generating adversarial training samples and training based on contrastive learning.

4.3 Experiment Setting

We run the baseline models with the default settings as reported in their original papers. We implement our model `SEAGEN` in Python 3.8 and run it on a NVIDIA A100 GPU. Datasets are split into training and test sets with a split ratio of 8:2 without overlapping. A pre-trained BERT model is used to compute textual embeddings as the initial graph node features of `SEAGEN`. The weighting parameter γ in the final loss function is set as $5e-5$ via a grid search in $\{5e-3$, $5e-4$, $5e-5$, $5e-6\}$. The model parameters are optimised using the Adam algorithm, and the model performance is evaluated by a 5-fold cross validation. The average accuracy and F1 scores are reported as the evaluation metrics. We will release our code in: https://github.com/gszswork/SEAGEN

Table 2: Fake News Detection Performance on `Twitter` and `FakeNewsNet` (F-F1: F1 score for fake news detection; T-F1: F1 score for true news detection).

Method	`Twitter`			`FakeNewsNet`		
	Acc	F-F1	T-F1	Acc	F-F1	T-F1
RvNN	0.805	0.803	0.807	0.828	0.801	0.829
Sta-PLAN	0.780	0.780	0.779	0.800	0.794	0.801
STS-NN	0.834	0.834	0.833	0.858	0.857	0.858
Bi-GCN	0.864	0.865	0.863	0.889	0.889	0.890
Dy-GCN	0.873	0.872	0.873	0.896	0.894	0.896
GACL	0.878	0.875	0.880	0.905	0.906	0.902
`SEAGEN`	**0.908**	**0.910**	**0.906**	**0.930**	**0.929**	**0.931**
gain	+3%	+3.5%	+2.6%	+2.5%	+2.3%	+2.9%

4.4 Performance Comparison

Table 2 shows the overall model performance results. On the two datasets, our model `SEAGEN` significantly outperforms all the competitors – the performance gain is up to 3%. This confirms the effectiveness of `SEAGEN` and using the temporally evolving graph embeddings for fake news detection.

Further, we observe that the methods using graph neural networks (Bi-GCN, Dy-GCN, GACL) outperform sequence-based methods (RvNN, Sta-PLAN, and STS-NN), which confirms the effectiveness of graph based methods.

Meanwhile, the dynamic graph method Dy-GCN learns from snapshots of static graphs at different time points. It only yields a marginal improvement over Bi-GCN. Its performance is limited by its coarse-grained graph encoding because the temporal graph information between snapshots cannot be captured. GACL which is an enhanced Bi-GCN model achieves the second best performance. We attribute this to the adversarial training samples and contrastive learning.

4.5 Ablation Study

To analyse the impact of each module in `SEAGEN`, we implement the following variants:

- **w/o LSGE**: Removing the local sub-graph encoder and feeding the node features in temporal order directly into the Global Evolution Capturing module.
- **w/o GEC**: Removing the global evolution capturing module.
- **w/o Hawkes**: Removing the neural Hawkes process module and deactivating joint training.

The comparative performance patterns on the F1 scores are similar and hence are omitted due to space limit. Same below. Each variant is trained on the same datasets as the full model `SEAGEN`. The experimental results are shown in Table 3. We see that the full model `SEAGEN` outperforms all model variants, confirming that each module contributes to the overall model performance. Among

these variants, the w/o LSGE has the worst performance; this is because the SEAGEN will degrade to a pure sequence model without the LSGE kernel. We can also see that the GEC and Hawkes module do have contributions to the model performance by capturing evolution sequential and self-exciting features.

4.6 Early Detection Performance

To mitigate the negative impact of fake news, it is crucial to detect fake news as early as possible. Therefore, we further study our model performance on the early detection of fake news. We take the first 20%, 40%, 60%, 80%, and 100% (in terms of the response amount of a news) subsets of the two benchmark datasets to train and test the models and show the accuracy in Fig. 3. Here, we only show the most competitive baselines Dy-GCN and GACL to simplify the figure.

Table 3: Fake News Detection Accuracy on Twitter and FakeNewsNet.

Variant	Twitter	FakeNewsNet
w/o LSGE	0.866	0.884
w/o GEC	0.878	0.915
w/o Hawkes	0.899	0.918
SEAGEN	0.908	0.930

It can be observed that, with the change of subset amounts, our model SEAGEN consistently outperforms both Dy-GCN and GACL. As longer time spans are considered, the performance gaps increases because more detailed temporal information is provided. The performance gain of SEAGEN over GACL increases around 2% when only the first 20% replies/retweets of a source post is available, which can result from the temporal feature extraction in our modules. This confirms the effectiveness of SEAGEN in the early detection of fake news.

4.7 Case Study

We next showcase two samples from FakeNewsNet which are mis-classified by Bi-GCN, GACL and Dy-GCN but successfully classified by SEAGEN. The true news item A (Jennifer Aniston on a Friends Reunion: Anything Is a Possibility...I Mean, George Clooney Got Married.) and the fake news item B (Kristen Stewart

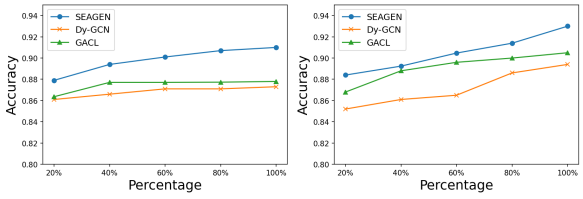

Fig. 3: Early Fake News Detection Performance on Twitter (left) and FakeNewsNet (right).

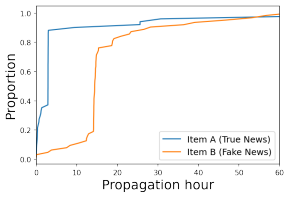

Fig. 4: ecdf plot for item A, B.

On Dating Robert Pattinson: The Public Were The Enemy.) share almost the same propagation structure where most relies and retweets straightly interact with the source post. Examples A and B also have similar propagation time spans (around 60 hours) and propagation size (around 70 responses), and hence they are difficult to be distinguished by static graph methods (Bi-GCN, GACL). However, their propagation speeds over time can vary. The propagation speeds of these two examples in the first 60 hours are visualised by their empirical Cumulative Distribution Function (ecdf) in Fig. 4. Compared to that of the true news item A, the propagation of fake news item B keeps mild until a viral spread hits at around 15 hours, which can be captured by `SEAGEN`'s GEC and the neural Hawkes module. In contract, Dy-GCN takes snapshots at equally spaced timestamps (e.g., 20 hours, 40 hours, 60 hours), thus failing to capture the sudden increase at early stage, which exemplifies the weakness of coarse-grained graph snapshot-based methods.

5 Related Work

Existing approaches for fake news detection can be broadly divided into *content-based*, *social context-based*, and *environment-based*.

Content-based approaches learn content or style features from the text or media content of news [4]. They may also leverage external knowledge for fact checking [15]. Social context-based approaches detect fake news through user features [17] or propagation analysis. The propagation analysis is a hot topic and develops from sequence modelling [8, 7] to graph modelling [1]. Propagation temporal features are also exploited recently, such as Choi et al. [2] encode the propagation as graph snapshots, Song et al.[18] utilises TGN [21] to encode the propagation graph, and [11] detects fake news through self-exciting difference. Environment-based approaches [12] mainly zooms out and considers the association across multiple news articles to debunk fake news.

In this paper, our approach focuses on social propagation graph, utilises graph and sequential features in a hybrid way. Self-exciting features is also captured by joint learning.

6 Conclusion

In this paper, we proposed a novel social media fake news detection model `SEAGEN`. It models the news propagation via self-exciting aware sub-graph sequence encodings. Extensive experiments on two benchmark datasets demonstrated the superiority of our model, including in detection in early stage. In the future work we intend to extend current work to user based temporal graph, which will be more suitable to be deployed as an online learning framework.

Acknowledgement

This research is supported by University of Melbourne and CSIRO.

References

1. Bian, T., Xiao, X., Xu, T., Zhao, P., Huang, W., Rong, Y., Huang, J.: Rumor detection on social media with bi-directional graph convolutional networks. In: AAAI (2020)
2. Choi, J., Ko, T., Choi, Y., Byun, H., Kim, C.k.: Dynamic graph convolutional networks with attention mechanism for rumor detection on social media. PLOS One **16**(8), e0256039 (2021)
3. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: NAACL (2019)
4. Feng, S., Banerjee, R., Choi, Y.: Syntactic stylometry for deception detection. In: ACL (2012)
5. Hawkes, A.G.: Spectra of some self-exciting and mutually exciting point processes. Biometrika **58**(1), 83–90 (1971)
6. Huang, Q., Zhou, C., Wu, J., Liu, L., Wang, B.: Deep spatial–temporal structure learning for rumor detection on twitter. Neural Computing and Applications (2020)
7. Khoo, L.M.S., Chieu, H.L., Qian, Z., Jiang, J.: Interpretable rumor detection in microblogs by attending to user interactions. In: AAAI (2020)
8. Ma, J., Gao, W., Mitra, P., Kwon, S., Jansen, B.J., Wong, K.F., Cha, M.: Detecting rumors from microblogs with recurrent neural networks. In: IJCAI (2016)
9. Ma, J., Gao, W., Wong, K.F.: Detect rumors in microblog posts using propagation structure via kernel learning. In: ACL (2017)
10. Ma, J., Gao, W., Wong, K.F.: Rumor detection on twitter with tree-structured recursive neural networks. In: ACL (2018)
11. Naumzik, C., Feuerriegel, S.: Detecting false rumors from retweet dynamics on social media. In: WWW (2022)
12. Nguyen, V.H., Sugiyama, K., Nakov, P., Kan, M.Y.: Fang: Leveraging social context for fake news detection using graph representation. In: Proceedings of the 29th ACM international conference on information & knowledge management. pp. 1165–1174 (2020)
13. Nie, H.R., Zhang, X., Li, M., Dolgun, A., Baglin, J.: Modelling user influence and rumor propagation on twitter using hawkes processes. In: DSAA (2020)
14. Robert, C.P., Casella, G., Casella, G.: Monte Carlo statistical methods, vol. 2. Springer (1999)
15. Samarinas, C., Hsu, W., Lee, M.L.: Improving evidence retrieval for automated explainable fact-checking. In: NAACL (2021)
16. Shu, K., Mahudeswaran, D., Wang, S., Lee, D., Liu, H.: Fakenewsnet: A data repository with news content, social context, and spatiotemporal information for studying fake news on social media. Big data **8**(3), 171–188 (2020)
17. Shu, K., Wang, S., Liu, H.: Beyond news contents: The role of social context for fake news detection. In: WSDM (2019)
18. Song, C., Shu, K., Wu, B.: Temporally evolving graph neural network for fake news detection. Information Processing & Management **58**(6), 102712 (2021)
19. Sun, T., Qian, Z., Dong, S., Li, P., Zhu, Q.: Rumor detection on social media with graph adversarial contrastive learning. In: WWW (2022)
20. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: NeurIPS (2017)
21. Xu, D., Ruan, C., Korpeoglu, E., Kumar, S., Achan, K.: Inductive representation learning on temporal graphs. ICLR (2020)
22. Zuo, S., Jiang, H., Li, Z., Zhao, T., Zha, H.: Transformer hawkes process. In: ICML (2020)