Reinforcement Subgraph Reasoning for Fake News Detection

Ruichao Yang
Hong Kong Baptist University
Hong Kong SAR, China
csrcyang@comp.hkbu.edu.hk

Xiting Wang*
Microsoft Research Asia
Beijing, China
xitwan@microsoft.com

Yiqiao Jin
University of California, Los Angeles
Los Angeles, United States
ahren2040@g.ucla.edu

Chaozhuo Li
Microsoft Research Asia
Beijing, China
cli@microsoft.com

Jianxun Lian
Microsoft Research Asia
Beijing, China
jianxun.lian@microsoft.com

Xing Xie
Microsoft Research Asia
Beijing, China
xing.xie@microsoft.com

ABSTRACT

The wide spread of fake news has caused serious societal issues. We propose a subgraph reasoning paradigm for fake news detection, which provides a crystal type of explainability by revealing which subgraphs of the news propagation network are the most important for news verification, and concurrently improves the generalization and discrimination power of graph-based detection models by removing task-irrelevant information. In particular, we propose a reinforced subgraph generation method, and perform fine-grained modeling on the generated subgraphs by developing a Hierarchical Path-aware Kernel Graph Attention Network. We also design a curriculum-based optimization method to ensure better convergence and train the two parts in an end-to-end manner. Extensive experiments show that our model outperforms the state-of-the-art methods and demonstrate the explainability of our method.

CCS CONCEPTS

• **Computing methodologies** → **Neural networks**.

KEYWORDS

Subgraph Reasoning, Fake News, Explainability, Social Network

ACM Reference Format:
Ruichao Yang, Xiting Wang, Yiqiao Jin, Chaozhuo Li, Jianxun Lian, and Xing Xie. 2022. Reinforcement Subgraph Reasoning for Fake News Detection. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22), August 14–18, 2022, Washington, DC, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3534678.3539277

1 INTRODUCTION

As social media facilitates efficient information sharing, it has also proven to be Petri dishes for fake news [4, 43]. The rapid, unconstrained spread of fake news becomes a global societal issue and has caused serious consequences, such as damaging public health, negatively impacting environment protection, and even causing death [37]. Meanwhile, people often find it hard to intercept fake news due to their deliberately fabricated content and complex dissemination patterns [4, 8], which highlights the importance of automatic detection of fake news.

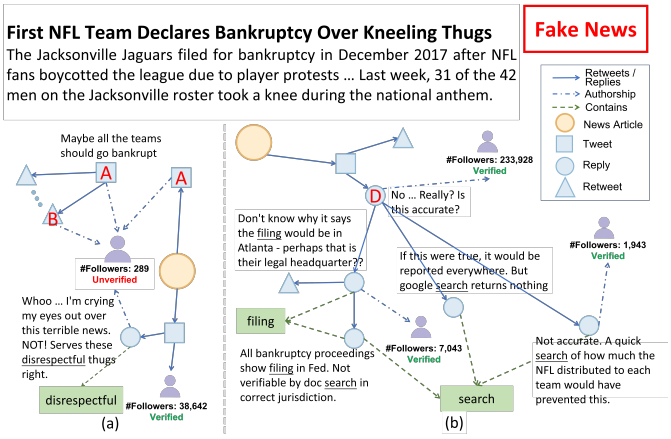

Figure 1: A motivating example for subgraph reasoning.

Recently, researchers have found that the news propagation structure on social networks provides crucial information for detecting fake news [28]. Methods have been proposed to model the propagation network based on neural networks such as Recurrent Neural Networks (RNNs) or Graph Neural Networks (GNNs) [12, 28, 54]. While these works have shown the effectiveness of propagation networks in improving the **accuracy** of fake news detection, they overlook the capability of propagation networks in enhancing **explainability**. Explainability is crucial for stopping the spread of fake news detection. Research on human cognition has found that misinformation continues to spread even if it has been detected, as people keep on believing in debunked falsehoods [20]. Thus, it is very important to educate people and provide deep insights for understanding the propagation of fake news [37]. While explainability is critical for insight discovery, it is currently under-explored. Pioneering efforts in explainable fake news detection either show which high-level statistics about the propagation networks (e.g., community density) are important for fake news detection [54], or provide simple, list-wise explanations, like a list of important posts on social media [15, 39]. Since the detailed information about topological connections are lost, it is difficult for them to provide a deep insight with convincing details, and answering fundamental questions like what propagation structures are the key indicators for fake news.

*Xiting Wang is the correspondence author.

KDD '22, August 14–18, 2022, Washington, DC, USA.

ACM ISBN 978-1-4503-9385-0/22/08…$15.00
https://doi.org/10.1145/3534678.3539277

To address these issues, we propose a novel paradigm called **subgraph reasoning**. As shown in Fig. 1, subgraph reasoning provides a crystal type of interpretability by explaining about which subgraphs in the propagation network are important for fake news detection. The key subgraphs are supposed to contain the most important propagation edges (e.g., reply or retweet) as well as the associated nodes (e.g., posts and users), which form a group of connected evidence for news verification. These subgraphs endow a deep reasoning about how fake news propagates, as well as how we may detect them. For example, Fig. 1(a) shows an interesting propagation pattern of fake news: an unverified user writes posts to attack the major organization involved in the fake news in diversified ways, including posting original tweets (A), retweeting a post written by himself (B), and replying tweets posted by celebrities (C). His tweets eventually trigger many retweets. Fig. 1(b) shows another subgraph important for verifying that the news is fake: a celebrity with many followers posts a tweet to question the authenticity of the news (D), which triggers many replies that involve proofs for fake news. These proofs are interconnected with clues like "filing" and "search". The reply hierarchy is relatively deep and involves multiple verified users with a relatively large number of followers.

In addition to explainability, subgraph reasoning also provides opportunities to further enhance the generalization and discrimination power of the detection model. First (generalization), propagation networks often contain many task-irrelevant nodes and edges, for example, posts that are not discussing about the authenticity of the news article [15]. When applying graph neural networks on such noisy structures, task-irrelevant information will be mixed into a node's neighborhood, which is known to reduce the generalization power of the subsequent classifier [53]. Detecting fake news based on key subgraphs alleviates this problem by removing potentially task-irrelevant nodes and edges. Second (discrimination), removing a large number of irrelevant nodes enables us to perform fine-grained modeling with an acceptable GPU memory cost. For example, we can determine the information to be propagated in the graph by carefully comparing every pair of features (e.g., tokens) in every pair of nodes. Such fine-grained feature-level modeling has been proven effective in extracting subtle clues that are essential for fact verification [15, 25], but is usually applied only on small graphs (e.g., with 5 nodes) due to limited GPU memory. Generating subgraphs enables us to perform fine-grained modeling with high discrimination power on large and noisy propagation networks.

While subgraph reasoning is promising, it also poses three major technical challenges. The first challenge pertains to subgraph generation (C1). There are a super-exponential number of candidate subgraphs and no ground-truth labels for key subgraphs. Thus, it is challenging to identify high-quality key subgraphs in tractable time. Second, how to perform fine-grained modeling on multiple heterogeneous subgraphs remains unclear (C2). Current fine-grained, feature-level graph neural networks typical focus on a single homogeneous graph [25]. It is unclear how they can be extended to thoroughly model heterogeneous connections between multiple types of nodes (e.g., posts and users), or modeling inter-subgraph and intra-subgraph connections in a unified manner. The last challenge is about joint optimization (C3), namely, how does one jointly optimize subgraph generation and fine-grained modeling in an end-to-end framework.

We solve these challenges by proposing a Subgraph reasoning framework for fake news detection (**SureFact**)[1]. The framework contains two major parts. In the first part, subgraph generation, a policy network is designed to effectively search candidate solutions based on both the topology information and features of a node (C1). In the second part, we design a Hierarchical Path-Aware Kernel Graph Attention Network (HP-KGAT) to accurately detect fake news based on multiple heterogeneous subgraphs (C2). Finally, we propose a curriculum-based optimization method that gradually enables end-to-end joint training of the two parts with the help of heuristically constructed pseudo labels (C3).

In summary, we make the following technical contributions.

- We propose a novel subgraph reasoning framework for fake news detection, which achieves superior explainability and improves accuracy by enhancing the generalization and discrimination power of fake news detection models.
- We design a Hierarchical Path-Aware Kernel Graph Attention Network, which accurately detects fake news by performing fine-grained modeling on multiple heterogeneous subgraphs.
- We introduce a curriculum-based optimization method that ensures convergence to a better solution by gradually increasing the learning difficulty and ensuring end-to-end training.

2 METHOD

2.1 Problem Formulation

Given a propagation network G of a news article, subgraph modeling predicts a label y for the news, and outputs a set of subgraphs $g_{1\sim M}$ that are important for the prediction.

Model input. The input **propagation network** is represented by $G = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V}$ denotes the node set and $\mathcal{E}$ is the edge set. Following the paradigm of fact verification [25], we consider two parts of data: claims from the news article, and evidence from the social media for verifying the claims, as shown in Fig. 2(a).

Claims to be verified are the news article content. We represent the claims at two levels to reason with both the overall context and key details. The document-level claim $d \in C$ is a node that denotes the entire news with both the news title and body (Fig. 2(a)). A sentence-level claim $c \in C$ is a node that represents either the news title or a sentence in the news body. d is connected with sentence-level claims with the edge type *title* or *body*.

Evidence for verifying the claims is from Twitter. We construct the evidence graph by following the classic schema in microblog retrieval [23]. In particular, three major types of evidence nodes are social posts, users, and keywords. The **posts** P include tweets, replies for tweets, or retweets about the news article, which are obtained from the fake news dataset [40]. The posts are connected with the *reply* or *retweet* relationship. The **users** U are represented by their features in the dataset, e.g., number of followers, as well as pattern-driven features in [54], e.g., the number of times a user involves in the discussion of fake news. Users are also connected with *reply* or *retweet* relations. The **keywords** in K are extracted with topic models and connected with *co-occurrence* by following Jin et al. [15]. The four types of nodes, claims, posts, users, and keywords, are **inter-connected** according to the relations shown in

[1] Codes and data: https://anonymous.4open.science/r/SureFact-Submission-E38E/

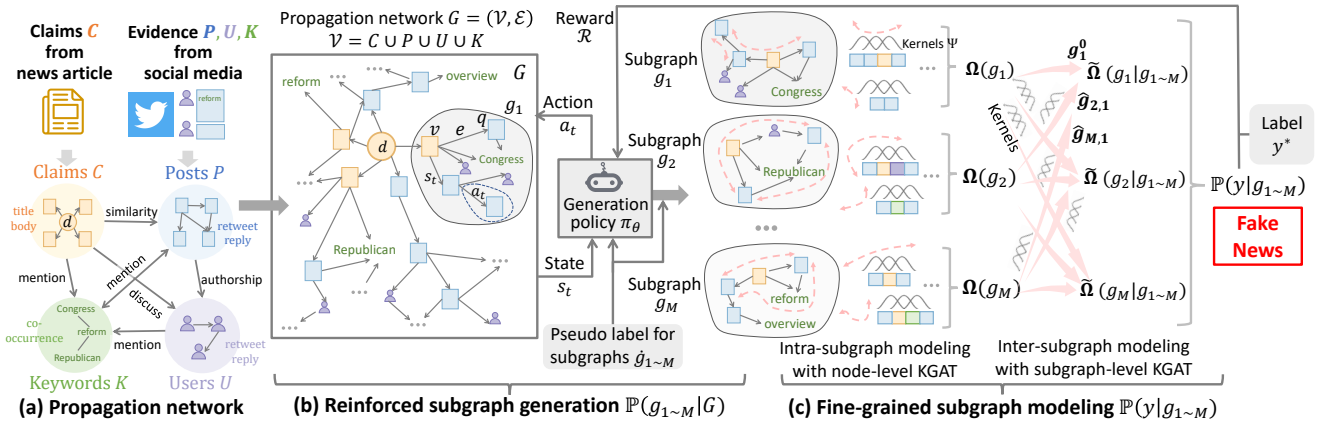

Figure 2: Our proposed *SureFact* framework for fake news detection.

Fig. 2(a). More details for constructing the propagation network G are given in the supplement.

Model output. Given a propagation network G, we output

- A **label** y for the news, which can be fake ($y = 1$) or real ($y = 0$).
- A set of **subgraphs** $g_{1\sim M} = \{g_1 ..., g_M\}$, which serves as the reason for predicting the label y. Here, each $g_m \subset G$ is a connected subgraph of G that is important for fake news detection.

2.2 Subgraph Reasoning Framework

To predict the label y for a news article through subgraphs, we propose a *Subgraph reasoning framework for Fake News Detection* (*SureFact*). Mathematically, the framework is formulated as

$$\mathbb{P}(y \mid G) \approx \sum_{\substack{g_{1\sim M}=\{g_1,...,g_M\} \\ g_m \subset G}} \underbrace{\mathbb{P}(y \mid g_{1\sim M})}_{\text{Subgraph Modeling}} \underbrace{\mathbb{P}(g_{1\sim M} \mid G)}_{\text{Subgraph Generation}} \tag{1}$$

Accordingly, our framework contains two modules (Fig. 2):

The **reinforced subgraph generation** module estimates the probability distribution $\mathbb{P}(g_{1\sim M} \mid G)$ of important subgraphs by maximizing the expected accuracy for fake news detection. This module enables extracting the key parts of G in the absence of ground-truth labels for subgraphs. Moreover, it filters noise and makes fine-grained modeling at the subsequent step possible.

The **fine-grained subgraph modeling** module learns $\mathbb{P}(y \mid g_{1\sim M})$ with our proposed *Hierarchical Path-aware Kernel Graph ATtention networks (HP-KGAT)*, which predicts the label y by effectively integrating the heterogeneous path information and jointly modeling both intra-subgraph and inter-subgraph relations.

Finally, a **curriculum-based optimization** method is developed to effectively optimize the two modules in an end-to-end way. The core is to gradually increase the learning difficulty and ensure end-to-end refinement for both modules.

2.3 Reinforced Subgraph Generation

The key for generating subgraphs is to learn $\mathbb{P}(g_{1\sim M} \mid G)$ without ground-truth labels for subgraphs. We propose to generate subgraphs by maximizing the expected accuracy for fake news detection, which can be achieved by reinforcement learning (RL). Let us denote the reward function that measure the accuracy for a prediction y as $\mathcal{R}^*(y)$. The reason why we can use RL to directly optimize the first module (subgraph generation) is illustrated as follows:

$$\sum_{y\in\{0,1\}} \mathbb{P}(y \mid G)\mathcal{R}^*(y) \approx \sum_{g_{1\sim M}\subset G} \mathbb{P}(g_{1\sim M} \mid G)\mathcal{R}(g_{1\sim M}), \tag{2}$$

$$\mathcal{R}(g_{1\sim M}) = \sum_{y\in\{0,1\}} \mathbb{P}(y \mid g_{1\sim M})\mathcal{R}^*(y) \tag{3}$$

where Eq. (2) can be easily derived based on Eqs. (1)(3) (a proof given in the supplement). In Eq. (2), the left side denotes the expected accuracy, and the right side represents the expected reward for subgraph generation. It shows that when we set the reward $\mathcal{R}$ for subgraph generation according to Eq. (3), we can optimize the subgraph generation module with RL (maximize the right side) and achieve approximate optimal expected accuracy for fake news detection (maximize the left side). Next, we introduce our reinforced subgraph generation method for optimizing reward $\mathcal{R}$. We first show how to generate **one subgraph** ($M = 1$) given a predefined seed node x for starting the generation. In particular, we define the Markov Decision Process (Sec. 2.3.1) and design a policy network for subgraph generation (Sec. 2.3.2). We then introduce how to extend the method for generating **multiple subgraphs** ($M > 1$) when no predefined seed nodes are given (Sec. 2.3.3).

2.3.1 Markov Decision Process (MDP). To maximize the expected reward by using RL, we need to formulate the problem as an MDP. An MDP is a 4-tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R})$ [42], where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ is the state transition function, and $\mathcal{R}$ is the reward function.

State. The state s_t reflects the status of the subgraph generation process at time t, and is defined as $s_t = (d, \chi_t, \mathcal{V}_t)$, where d is the news document, χ_t is the subgraph generated at time t, and $\mathcal{V}_t$ denotes the newly absorbed nodes. The initial state $s_0 = (d, \{x\}, \{x\})$, where x is the seed node for generation.

Action. An action a_t at time t is defined as $a_t = \mathcal{V}_{t+1}$, where $\mathcal{V}_{t+1}$ is the set of nodes to be absorbed if action a_t is taken. The candidate nodes to be absorbed $\mathcal{Q}_{t+1}$ are the neighbors of $\mathcal{V}_t$, except the historically visited ones: $\mathcal{Q}_{t+1} = \{q \mid (v, e, q) \in G, v \in \mathcal{V}_t, q \notin \chi_t\}$. To speed up the generation process, we constrain the action space by following Liu et al. [22]. Specifically, a hyper-parameter *receptive field size* ($N_{r,t}$) is set for each time t, which constrains the number of nodes to be absorbed for each node in $\mathcal{V}_t$. A self-loop is added

to every node, and selecting the self-loops equals to absorbing a number of nodes that is fewer than the given $N_{r,t}$.

Transition. Given state s_t and action a_t, the environment deterministically transits into the next state s_{t+1}: $\mathbb{P}(s_{t+1} = (d, \chi_{t+1}, \mathcal{V}_{t+1}) \mid s_t, a_t) = 1$. Subgraph χ_{t+1} is built by adding newly absorbed nodes $\mathcal{V}_{t+1}$ to subgraph χ_t, as well as all edges between $\mathcal{V}_{t+1}$ and χ_t.

Reward. We consider only the terminal reward at the last time point T, when the subgraph(s) $g_{1\sim M}$ has already been generated. We set the reward function $\mathcal{R}(g_{1\sim M})$ according to Eq. (3), so that solving the MDP with RL methods is equal to maximizing the expected accuracy for fake news detection. Here, $\mathcal{R}^*(y) = \mathbb{I}(y = y^*)$, which is 1 (or 0) if y is (or is not) equal to the ground-truth label y^*.

2.3.2 Subgraph Generation Policy. The generation policy $\pi_\theta(a_t, s_t)$ outputs the probability for taking action a_t given the state s_t:

$$\pi_\theta\left(a_t = \mathcal{V}_{t+1}, s_t\right) = \prod_{v \in \mathcal{V}_{t+1}} \pi'_\theta\left(v, s_t\right) \tag{4}$$

The nodes to be selected are modeled independently, so that we can select nodes with descending π'_θ to achieve the optimal result. π'_θ is defined by considering both the topology and node attributes:

$$\pi'_\theta\left(v, s_t\right) = \frac{\exp(W_0 \operatorname{ReLU}\left(W_1\left(\mathbf{o}_v \oplus \mathbf{z}_v \oplus \mathbf{s}_t\right)\right))}{\sum_{q \in \mathcal{Q}_{t+1}} \exp(W_0 \operatorname{ReLU}\left(W_1\left(\mathbf{o}_q \oplus \mathbf{z}_q \oplus \mathbf{s}_t\right)\right))} \tag{5}$$

where $\oplus$ is the concatenation operator, $\mathbf{o}_v$ and $\mathbf{z}_v$ are the topological and attribute embeddings of node v, and $\mathbf{s}_t$ is the state embedding. We learn the topological embedding $\mathbf{o}_v$ by using TransE [1]. When v is a textual node (claim, post or keyword), $\mathbf{z}_v$ is computed based on their BERT representations [7]. When v is a user, $\mathbf{z}_v$ is computed by concatenating v's feature embeddings, each obtained through a look up layer. The embeddings of different types of nodes go through a different fully connected layer before assigning to $\mathbf{z}_v$, to ensure that they are mapped into the same space. The state embedding $\mathbf{s}_t$ of s_t is

$$\mathbf{s}_t = \mathbf{d} \oplus \boldsymbol{\chi}_t \oplus \mathbf{V}_t \tag{6}$$

where $\mathbf{d}$ is the BERT embedding of the news article, $\boldsymbol{\chi}_t$ is the representation of the generated subgraph χ_t, and $\mathbf{V}_t$ is the joint representation of the newly added nodes V_t. To derive $\boldsymbol{\chi}_t$, the representation of each node v in the extracted subgraph χ_t is first refined by aggregating information from its neighbors following [22]:

$$\mathbf{z}'_v = \tanh(W_2(\mathbf{o}_v \oplus \mathbf{z}_v \oplus \sum_{(v,e,q) \in \chi_t} (\mathbf{o}_e + \mathbf{o}_q) \oplus \mathbf{z}_q)) \tag{7}$$

where q is a neighbour of v in χ_t and e is their in-between edge. Then, $\boldsymbol{\chi}_t$ is calculated by aggregating $\mathbf{z}'_v$ with an attention layer:

$$\boldsymbol{\chi}_t = \textstyle\sum_{v \in \chi_t} \alpha_v \mathbf{z}'_v, \quad \alpha_v = \dfrac{\exp(W_3 \operatorname{ELU}(W_4 \mathbf{z}'_v))}{\sum_{q \in \chi_t} \exp(W_3 \operatorname{ELU}(W_4 \mathbf{z}'_q))} \tag{8}$$

where ELU is the exponential linear unit. The representation of the newly added nodes $\mathbf{V}_t$ is the average of node embeddings in $\mathcal{V}_t$:

$$\mathbf{V}_t = W_5 \operatorname{ELU}(W_6 \frac{\sum_{v \in \mathcal{V}_t} \mathbf{o}_v \oplus \mathbf{z}_v}{|\mathcal{V}_t|}) \tag{9}$$

Given a sequence of actions $a_0, a_1 ..., a_T$ and states $s_0, s_1 ..., s_T$ that lead to the generation of $g_{1\sim M}$, we can estimate the probability of generating the subgraphs based on the policy network:

$$\mathbb{P}(g_{1\sim M} | G) = \prod_{t \in [0,T]} \pi_\theta(a_t, s_t) \tag{10}$$

We introduce how the policy network can be effectively optimized to achieve the maximum expected reward in Sec. 2.5.

2.3.3 Generating Multiple Subgraphs. We can generate multiple subgraphs without predefined seed nodes by carefully designing G and slightly modifying the MDP. In particular, we start the generation process from node d that represents the entire news, i.e., the initial state $s_0 = (d, \chi_0 = \{d\}, \mathcal{V}_0 = \{d\})$. By designing G so that d is connected with all candidate seed nodes (in our implementation news claims and tweets) and setting the receptive field size at time $t = 1$ to M, we can use the policy network to automatically obtain M seed nodes at time $t = 1$. All the nodes absorbed when $t > 1$ can be assigned to one seed node, each corresponds to a subgraph. The overlaps between M subgraphs are automatically avoided, since no nodes will be selected twice.

2.4 Fine-Grained Subgraph Modeling

After generating subgraphs $g_{1\sim M}$, we conduct fine-grained modeling on the subgraphs to detect fake news. The key is to accurately estimating $\mathbb{P}(y|g_{1\sim M})$ by effectively modeling multiple heterogenous subgraphs. We achieve this goal by extending Kernel Graph Attention Network (KGAT) [25], which utilizes a kernel-based attention to effectively concentrate on meaningful subtle clues, and has been proved effective in fact verification [25]. However, it is designed for a single graph with one type of nodes (textual). We extend KGAT to *Hierarchical Path-aware Kernel Graph ATtention networks (HP-KGAT)*, which integrates heterogeneous path information, and applies the kernel attention hierarchically in both the node and subgraph levels to jointly model inter-sugraph and intra-subgraph connections (Fig. 2). Next, we introduce how we model the intra-subgraph relations (Sec. 2.4.1), how to model the inter-subgraph relations (Sec. 2.4.2), and how to combine the two levels to estimate the probability distribution $\mathbb{P}(y|g_{1\sim M})$ of the label y (Sec. 2.4.3).

2.4.1 Intra-Subgraph Modeling with Node-Level KGAT. As shown in Fig. 2(c), intra-graph modeling outputs a subgraph representation $\Omega(g_m)$ for each subgraph g_m by effectively propagating information *within* the heterogeneous subgraph. Specifically, our intra-subgraph modeling consists of the following steps.

Step 1. Node representation initialization. The representation of each node v is initialized by using its attribute embedding $\mathbf{z}_v$ (Sec. 2.3.2). In addition, each word token in a textual node is considered as a candidate subtle clue, for which we compute an embedding by using BERT. Specifically, the hidden states $\mathbf{h}_v^1, \mathbf{h}_v^2, ...\mathbf{h}_v^{|v|}$ at the last layer of BERT are used as the token embeddings, where $\mathbf{h}_v^i$ denotes the embedding for the i-th token in node v.

Step 2. Kernel match features between nodes. For each pair of textual nodes v, q, their fine-grained match features are extracted by constructing a translation matrix $L_{q,v}$. An entry in $L_{q,v}$ is the cosine similarity between the token representations: $L_{q,v}^{i,j} = \cos(\mathbf{h}_q^i, \mathbf{h}_v^j)$. Based on the translation matrix, Υ kernels are used to extract the kernel match features between the i-th token in q and node v:

$$\Psi_\tau(L_{q,v}^i) = \log \textstyle\sum_j \exp(-\frac{(L_{q,v}^{i,j} - \mu_\tau)^2}{2\sigma_\tau^2}) \tag{11}$$

$$\vec{\Psi}(L_{q,v}^i) = \{\Psi_1(L_{q,v}^i), \ldots, \Psi_\Upsilon(L_{q,v}^i)\} \tag{12}$$

Each Ψ_τ is a Gaussian kernel [6, 47], which counts the expected number of similarity scores that fall into the region defined by mean μ_τ and standard deviation σ_τ. By combining Υ kernels, $\vec{\Psi}(L^i_{q,v})$ summarizes how matched the i-th token in q is with all tokens in v at different similarity levels. Such kernel match features have been shown effective in modeling subtle clues and verifying facts [15, 25].

Step 3. Path-aware token-level attention. KGAT computes the fine-grained information to be propagated from textual node q to v by using a token-level attention layer. We extend the attention so that it not only considers token-level kernel match features, but also models all heterogeneous paths between q and v to effectively integrate topological information. Such topological information provides crucial information for fake news detection. For example, given a post "I doubt it", it is important to know which post it *replies* (edge) to correctly understanding whether it supports the news is true. For two posts, it is important to know whether there are susceptible users (connected nodes) involved in the discussion. By designing a path-aware token-level attention, the model can extract subtle clues (important tokens) by carefully considering such information. Specifically, given the k-th path between nodes q and v, we compute the attention score $\alpha^{i,k}_{q,v}$ for the i-th token in q by

$$\alpha^{i,k}_{q,v} = \text{softmax}_i(W_7\vec{\Psi}(L^i_{q,v}) + W_8\zeta^k_{q,v} + b_1) \quad (13)$$

$$\widehat{\mathbf{z}}_{q,v} = \frac{1}{K}\sum_i\sum_k \alpha^{i,k}_{q,v}(\mathbf{h}^i_q \oplus \zeta^k_{q,v}) \quad (14)$$

where K is the total number of paths between q and v in g_m, $\zeta^k_{q,v}$ is the embedding of the k-th path, and $\widehat{\mathbf{z}}_{q,v}$ denotes the fine-grained information to be propagated from node q to v. We derive the path embedding $\zeta^k_{q,v}$ by concatenating the embeddings of its edges and nodes, and encode the concatenated vector through a multilayer perceptron: $\zeta^k_{q,v} = \text{MLP}(\oplus_e \mathbf{o}_e \oplus_{v'} \mathbf{z}_{v'})$, where e and v' are an edge and a node in the path, respectively. We pad zeros in the concatenated vector to ensure that they are of the same length.

Step 4: Node-level attention. We then derive a final representation κ_v for node v by attentively integrating $\widehat{\mathbf{z}}_{q,v}$ into $\mathbf{z}_v$:

$$\gamma_{q,v} = \text{softmax}_q(\text{MLP}(\widehat{\mathbf{z}}_{q,v} \oplus \mathbf{z}_v)) \quad (15)$$

$$\kappa_v = (\textstyle\sum_{q\in g_m \setminus U} \gamma_{q,v} \cdot \widehat{\mathbf{z}}_{q,v}) \oplus \mathbf{z}_v \quad (16)$$

The kernel-based node representation κ_v aggregates token-level subtle clues as well as the heterogeneous path information from every textual node in g_m. We then attentively aggregate the kernel-based node representations to get a subgraph embedding $\Omega(g_m)$:

$$\Omega(g_m) = \sum_{v\in g_m\setminus U} \gamma'_v\kappa_v, \;\; \gamma'_v = \text{softmax}_v(W_9\,\text{average}_i\vec{\Psi}(L^i_{c,v})) \quad (17)$$

where $\vec{\Psi}(L^i_{c,v})$ is the kernel match features that estimate how matched the textual node v is with the news claims in g_m. Specifically, $L^i_{c,v}$ is a token-level translation matrix that measures cosine similarities between a token in the claims and a token in v. If g_m does not contain claims, we consider its claim as the entire news document d.

2.4.2 Inter-Subgraph Modeling with Subgraph-Level KGAT. As shown in Fig. 2(c), inter-subgraph modeling propagates information *between* subgraphs to derive a representation $\widehat{\Omega}(g_m|g_{1\sim M})$ for subgraph g_m. This is achieved by applying KGAT at the subgraph level. Specifically, we consider each subgraph g_m as a node in the traditional KGAT, and treat each node in g_m as a previous token. In this way, we can effectively model inter-graph relations by comparing every pair of nodes in given two subgraphs. Accordingly, the inter-subgraph modeling consists of the following steps.

Step 1. Subgraph representation initialization. For each subgraph g_m, its representation $\mathbf{g}_m$ is initialized by using BERT. Specifically, all the claims, posts, and keywords in g_m are concatenated by using the special token "[SEP]" and fed into BERT. The BERT embedding for the "[CLS]" token is then used as $\mathbf{g}_m$. Each node v in a subgraph g_m serves as a candidate subtle clue, whose embedding $\widehat{h}^v_m = \mathbf{z}_v$.

Step 2. Kernel match features between subgraphs. Given two subgraphs g_m and g_n, their kernel match features are calculated based on the translation matrix $\widehat{L}_{n,m}$. Each entry in $\widehat{L}_{n,m}$ is the cosine similarity between two nodes in different subgraphs: $\widehat{l}^{q,v}_{n,m} = \cos(\widehat{h}^q_n, \widehat{h}^v_m)$, where q is a node in g_n and v is a node in g_m. The kernel match feature $\vec{\Psi}(\widehat{L}^q_{n,m})$ is then derived according to Eqs. (11)(12).

Step 3. Node-level attention. Next, we compute a fine-grained representation $\widehat{\mathbf{g}}_{n,m}$, which denotes the information to be propagated from g_n to g_m, by using a node-level attention:

$$\rho^q_{n,m} = \text{softmax}_q(\text{MLP}(W_{10}(\vec{\Psi}(\widehat{L}^q_{n,m}) + b_2)) \quad (18)$$

$$\widehat{\mathbf{g}}_{n,m} = \textstyle\sum_{q\in g_n} \rho^q_{n,m}\widehat{h}^q_n \quad (19)$$

Step 4. Subgraph-level attention. We can then derive an enhanced subgraph representation $\widehat{\Omega}(g_m|g_{1\sim M})$ by attentively aggregating all related information $\widehat{\mathbf{g}}_{n,m}$ from other subgraphs:

$$\lambda_{n,m} = \text{softmax}_n\left(\text{MLP}\left(\widehat{\mathbf{g}}_{n,m} \oplus \mathbf{g}_m \oplus \mathbf{d}\right)\right) \quad (20)$$

$$\widehat{\Omega}(g_m|g_{1\sim M}) = \left(\textstyle\sum_m \lambda_{n,m}\cdot\widehat{\mathbf{g}}_{n,m}\right) \oplus \mathbf{g}_m \quad (21)$$

2.4.3 Joint Modeling. Given intra- and inter-subgraph representations $\Omega(g_m)$ and $\widehat{\Omega}(g_m|g_{1\sim M})$, we predict label y for the news with

$$\mathbb{P}(y \mid g_{1\sim M}) = \textstyle\sum_m \mathbb{P}(y \mid g_m)\cdot\mathbb{P}(g_m \mid g_{1\sim M}) \quad (22)$$

$$\mathbb{P}(y \mid g_m) = \text{sigmoid}(W_{11}(\Omega(g_m) \oplus \widehat{\Omega}(g_m|g_{1\sim M})) + b_3) \quad (23)$$

$$\mathbb{P}(g_m \mid g_{1\sim M}) = \text{softmax}_m(W_{12}\,\text{average}_i(\vec{\Psi}(\widehat{L}^i_{c,m})) + b_4) \quad (24)$$

where $\mathbb{P}(y \mid g_m)$ is a prediction of the label from the perspective of subgraph g_m, and $\mathbb{P}(g_m \mid g_{1\sim M})$ is the relative importance of g_m in fake news detection. Here, kernel match features $\vec{\Psi}(\widehat{L}^i_{c,m}))$ are derived by comparing the claims with evidence in g_m. In particular, $\widehat{L}_{c,m}$ is a node-level translation matrix that measures how similar a claim node in g_m is with an evidence node in g_m.

2.5 Curriculum-Based Optimization

Subgraph generation and fine-grained modeling are closely entangled. Ideally, the subgraph generation module would extract informative subgraphs that the fine-grained modeling module can utilize to verify the authenticity of the news story. However, as the generation policy π_θ is far from optimal at the beginning, the initially generated subgraphs may be uninformative and propagate noise to fine-grained subgraph modeling, leading to sub-optimal parameters in both modules. To tackle this challenge, we propose a curriculum-based optimization strategy. The basic idea of curriculum learning is to start from a simple task and gradually increasing the difficulty, which usually leads to a convergence to better solutions [29]. Following this idea, we first provide a simple task by offering direct supervision signals, i.e., pseudo labels for important

subgraphs (Sec. 2.5.1). The learning difficulty is low as the two modules are decoupled and are only required to perform supervised learning instead of RL. Then, we gradually increase the collaboration between the two modules with a curriculum-based joint optimization that finally leads to an end-to-end training (Sec. 2.5.2).

2.5.1 Pretraining with Pseudo Labels. We first introduce how to construct the pseudo labels for important subgraphs. Then, we explain how to pretrain each module with the pseudo labels.

Saliency-based pseudo label construction. We construct the pseudo labels $\dot{g}_{1\sim M}$ for subgraphs so that they contain important claims and evidence, and at the same time are diversified. To this end, we first compute a saliency score for each node in G by using an unsupervised algorithm, *mutual reinforcement ranking* [9], which extends PageRank [2] to better handle heterogeneous graphs. The basic idea is to propagate saliency scores through edges, so that the nodes (e.g., tweets) connected with (e.g., authored by) salient nodes (e.g., important users) are also considered salient. Mutual reinforcement ranking not only enables us to compute a reasonable saliency score based on G, but also increases the probability that salient nodes are connected with each other (approximately form subgraphs). After deriving the saliency scores, we then use diverse sibling beam search [21] to extract diversified subgraphs $\dot{g}_{1\sim M}$ which achieve an approximately maximum average saliency score.

Subgraph generation pretraining. Given pseudo labels $\dot{g}_{1\sim M}$, the subgraph generation policy π_θ is pretrained with behavior cloning [35], in which the pseudo labels are considered as ground-truth to guide the policy. This supervised paradigm is much easier compared with RL, which is usually difficult to converge when there are a super-exponential number of candidate solutions [22].

Fine-grained model pretraining. We pretrain the fine-grained modeling module so that it detects fake news based on the pseudo labels $\dot{g}_{1\sim M}$. This is achieved by minimizing the cross-entropy loss:

$$\mathcal{L}_{ce}(\dot{g}_{1\sim M}) = y^* \log(\mathbb{P}(y|\dot{g}_{1\sim M})) + (1-y^*)\log(1-\mathbb{P}(y|\dot{g}_{1\sim M})) \quad (25)$$

where y^* is the ground-truth label of the news.

2.5.2 Curriculum-Based Joint Optimization. Although pseudo labels can effectively guide model training, their unsupervised and heuristic nature can easily result in sub-optimal performance. Moreover, by using pseudo labels, the two modules are disconnected during training, which causes a discrepancy between training and testing. To solve these issues, we propose a method for jointly optimizing the two modules in an end-to-end way.

Subgraph generation optimization. After pretraining, the generation policy is trained to directly optimize the expected accuracy of fake news detection with reinforcement learning (Eq. (2)). Here, we use the widely-adopted RL method REINFORCE with baseline [42], which maximizes expected reward by minimizing

$$\mathcal{L}_{rl} = -(\mathcal{R}(\widehat{g}_{1\sim M}) - \mathcal{R}(\bar{g}_{1\sim M}))\log \mathbb{P}(\widehat{g}_{1\sim M}|G) \quad (26)$$

where $\widehat{g}_{1\sim M}$ is generated by sampling nodes based on the generation policy $\pi_\theta(a_t, s_t)$, and $\bar{g}_{1\sim M}$ is a baseline added to reduce variance, which is created by taking the action a_t that maximize $\pi_\theta(a_t, s_t)$.

Fine-grained model optimization. To further optimize the fine-grained modeling module, we conduct curriculum learning, in which $\dot{\mathbf{g}}_{1\sim M}$ and $\bar{\mathbf{g}}_{1\sim M}$ are treated as easy and hard material, respectively. We gradually shift from easy to hard material by minimizing

$$\mathcal{L}_{cr} = \varrho\mathcal{L}_{ce}(\dot{g}_{1\sim M}) + (1-\varrho)\mathcal{L}_{ce}(\bar{g}_{1\sim M}), \quad \varrho \sim \text{Bernoulli}(p_0) \quad (27)$$

where ϱ is sampled from a Bernoulli distribution parameterized by p_0. By gradually shrinking p_0 during training, the fine-grained modeling module is exposed to more hard materials $\bar{g}_{1\sim M}$, and eventually learns to collaborate with the generation policy.

3 EXPERIMENTS AND RESULTS

3.1 Experimental Setup

3.1.1 Dataset. We evaluate our method by using two public benchmark datasets, PolitiFact and GossipCop [40], which contain 815 and 7,612 news articles, perspectively. Each data sample in the datasets consists of a news article, its related posts on Twitter, metadata about the authors of the posts, and a label "real" or "fake" annotated by journalists and domain experts. We follow [10] to preprocess the data. Statistics of the datasets are given in the supplement.

3.1.2 Baselines. We compare with 11 baselines from two groups.

Methods in the first group (**G1**) detect fake news based on the textual content of the news and the external knowledge about the textual content, e.g., relations between news entities in the knowledge graphs. This group includes three methods. In particular, **B-TransE** [32] detects fake news based on knowledge graphs. **BERT-C** uses BERT [7], a large pretrained bidirectional Transformer, to predict a label for the news article content. We start with a pretrained BERT model and fine-tune it on our training set. The final prediction is made by the document-level representation "[CLS]". **RoBERTa-C** follows an identical fine-tuning procedure as BERT-C and uses RoBERTa [24] as its backbone.

Methods in the second group (**G2**) exploit social context of news in addition to the news content. Specifically, **DTC** [3] is a decision-tree-based method that classifies news based on users' posting and re-posting behaviors, as well as news content. **RFC** [19] detects fake news by using a random forest model and structural and linguistic features. **GCAN** [26] adopts Graph-aware Co-Attention Networks to model the interactions between the textual and social information. **dEFEND** [39] models both news content and user comments with a sentence-comment co-attention network. For **BERT-S** and **RoBERTa-S**, we follow Pelrine et al. [34] to first classify the social posts and convert the predictions of the posts to the prediction of the news through majority voting. **BERTweet** [30] is similar with BERT-S, but before fine-tuning, it is pretrained on 850 million tweets. **FinerFact** [15] uses bi-channel kernel graph network to capture subtle clues between evidence.

3.1.3 Evaluation Criteria. We adopt five widely-used criteria for evaluating the fake news detection performance: Precision (**Pre**), Recall (**Rec**), F1 score (**F1**), Accuracy (**Acc**), and Area under the ROC Curve (**AUC**). For each experiment, we conduct 5-fold cross-validation and report the average results.

3.1.4 Implementation Details. For the baselines, we set most of the hyperparameters by following the corresponding papers, and carefully tune important hyperparameters (e.g., learning rate and embedding size) to ensure optimal performance. We implement

Table 1: Comparison with baselines in terms of fake news detection performance. Best overall and baseline results are highlighted with bold and underlined text. G1 represents content-based methods, and G2 refers to social-based methods. Statistically significant improvement (t-test over 5 different dataset splits, p-value< 0.05) is marked by *.

		PolitiFact					GossipCop				
		Pre	Rec	F1	Acc	AUC	Pre	Rec	F1	Acc	AUC
G1	B-TransE	0.7739	0.7658	0.7641	0.7694	0.8340	0.7369	0.7330	0.7340	0.7394	0.7995
	BERT-C	0.8249	0.8465	0.8413	0.8327	0.8632	0.7863	0.7619	0.7628	0.8123	0.8371
	RoBERTa-C	0.8904	0.8784	0.8759	0.8843	0.9179	0.8134	0.8092	0.8276	0.8154	0.8623
G2	DTC	0.7397	0.7269	0.7362	0.7364	0.7618	0.7129	0.6794	0.6929	0.7156	0.7007
	RFC	0.7536	0.7450	0.7422	0.7582	0.8082	0.6978	0.6833	0.6534	0.6845	0.7366
	GCAN	0.8169	0.8208	0.8364	0.8372	0.8089	0.7824	0.8027	0.7757	0.7506	0.8066
	dEFEND	0.9016	0.8953	0.8879	0.8846	0.8904	0.7217	0.8015	0.7538	0.8098	0.8401
	BERT-S	0.8243	0.9013	0.8453	0.8619	0.9131	0.8459	0.8516	0.8542	0.8475	0.8533
	RoBERTa-S	0.8529	0.9052	0.8729	0.8804	0.9218	0.8441	0.8594	0.8397	0.8289	0.8625
	BERTweet	0.8440	0.9028	0.8658	0.8781	0.9155	0.8508	0.8624	0.8626	0.8478	0.8619
	FinerFact	0.9185	0.9043	0.9148	0.9088	0.9303	0.8609	0.8739	0.8657	0.8373	0.8644
Ours	SureFact	**0.9506***	**0.9283***	**0.9392***	**0.9436***	**0.9413***	**0.8796***	**0.8835***	**0.8811***	**0.8658***	**0.8797***

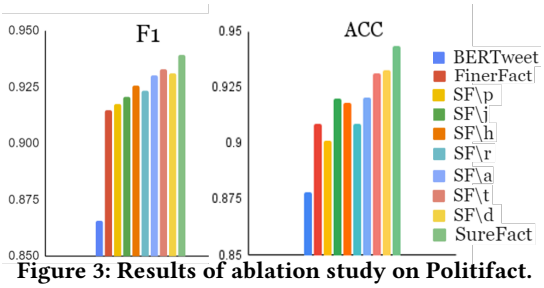

Figure 3: Results of ablation study on Politifact.

SureFact with PyTorch [33]. More implementation details are given in the supplementary material.

3.2 Detection Performance

We first evaluate whether our method can more accurately detect fake news compared with the state-of-the-art methods.

3.2.1 Overall Performance. Table 1 shows that our method performs better than all 11 baselines significantly (p-value< 0.05) in terms of five evaluation criteria on two benchmark datasets. For example, our method improves over the most competitive baseline, FinerFact, by 3.5% on PolitiFact and 3.4% on GossipCop in terms of accuracy. This indicates that subgraph reasoning can effectively improve the generalization and discrimination power of detection model. We also observe that social-based methods (G2) tend to outperform content-based methods (G1), This demonstrates the effectiveness of leveraging social data such as user comments for news verification. Within G2, we observe that methods based on fine-grained modeling of evidence (FinerFact) tends to achieve better performance than more advanced pretrained language models, which demonstrates the usefulness of fine-grained modeling. Among pretrained language models, BERTweet generally outperforms RoBERTa and BERT. This implies the usefulness of pretraining on domain-specific data (online posts) for better performance.

3.2.2 Ablation Study. We conduct an ablation study to demonstrate the effectiveness of each major component in *SureFact*. Specifically, we implement seven variants of our method: 1) **SF\p** removes the pretraining step in curriculum-based optimization; 2) **SF\j** eliminates the joint optimization step that enables end-to-end training; 3) **SF\h** removes the path-aware information of HP-KGAT during fine-grained subgraph modeling; 4) **SF\r** excludes the inter-subgraph embedding $\widehat{\Omega}(g_m|g_{1\sim M})$ from the final prediction; 5) **SF\a** eliminates the intra-subgraph embedding $\Omega(g_m)$; 6) **SF\t** excludes the node attribute information from policy generation policy; 7) **SF\d** replaces the diverse sibling beam search with traditional beam search. As shown in Fig. 3, removing each major component results in a decrease in detection accuracy, which illustrates their importance. Except for SF\p that removes pretraining with pseudo labels, all variants perform better than the most competitive baselines (FinerFact and BERTweet). This demonstrates that 1) pseudo-label-based pretraining is crucial for the success of subgraph reasoning and 2) our framework achieves a relatively robust improvement when a major component (except for pretraining) is removed.

3.2.3 Parameter sensitivity analysis. Our method consistently performs better than the baselines with different number of kernels, which demonstrates the robustness of SureFact (Appendix II).

3.3 Explainability

Our method uses only a very limited number of nodes (receptive field sizes $N_{r,1} \times N_{r,2} \times N_{r,3} = 64$) for detecting fake news. Thus, humans can easily understand which parts of the data the model uses with a small cognitive load. When the model is very accurate, the generated explanations not only enable the understanding of the model itself, but also provides insights about the dataset, e.g., which propagation structures (subgraphs) are the key indicators for detecting fake news. An interesting question is, given a user with a limited cognitive budget, how well s/he can obtain insight? To answer this question, we conduct an automatic quantitative experiment (Sec. 3.3.1) and a user study (Sec. 3.3.2).

3.3.1 Detection Performance v.s. Cognitive Load. In this experiment, we evaluate how the detection performance of different methods

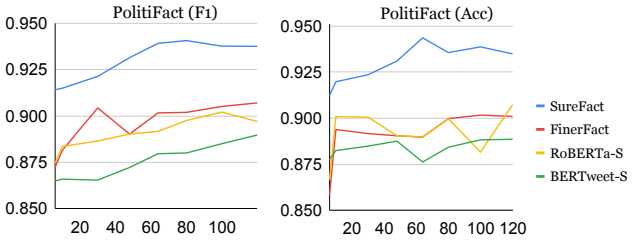

Figure 4: Detection performance v.s. cognitive load.

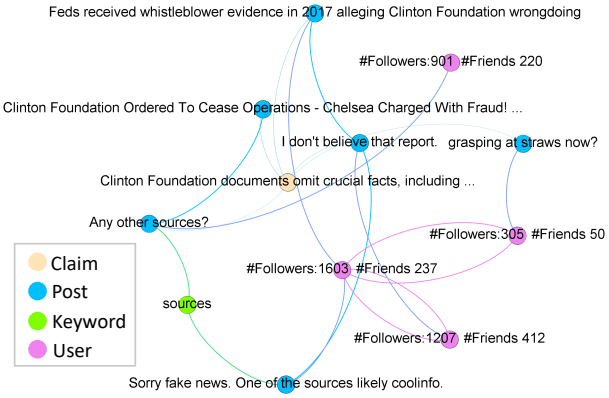

Figure 5: A subgraph generated by SureFact in the user study

change given different levels of cognitive load. We control the cognitive load for SureFact by setting the receptive field sizes, and filter the input data for baselines with mutual reinforcement ranking scores (Sec. 2.5.1). Fig. 4 shows the results of SureFact and the three most competitive baselines. We can see that our method outperforms the baselines at different levels of cognitive load, and achieves a good performance even when the cognitive load is relatively small. We can also see that increasing the number of input nodes does not necessarily lead to better performance. This aligns with the findings of [53], i.e., integrating task-irrelavant nodes into the neural networks may lead to overfitting.

3.3.2 User Study. To understand how much insight real-world users can obtain from the explanations, we conduct a user study. Specifically, we randomly sample 30 news articles from PolitiFact, including 15 fake news articles and 15 real ones. For each example, we generate 3 types of explanations. The first type is the explanations provided by **FinerFact**, which consists of a list of 30 posts and 30 users coming from 5 topics. The second type is the propagation network G (**PropagationNet**), which is visualized by using an open graph visualization platform Gephi[2]. We carefully adjust the graph to ensure there are no overlaps. The third type is the subgraphs generated by **SureFact**, which are also visualized by using Gephi. To ensure a fair comparison with FinerFact, we generate 5 subgraphs (correspond to 5 topics in FinerFact), each with 12 nodes. An example subgraph we generated is given in Fig. 5. We recruit 6 participants to label the explanations. For each news, a participant is shown one type of explanations and asked to 1) decide **whether the news is fake** within 2 minutes and 2) give a **confidence** score

[2]https://gephi.org/

Table 2: Results of the user study. Best results are highlighted in bold. Statistically significant improvement (t-test, p-value< 0.05) is marked by *.

	F1	Acc	Confidence	Agreement
FinerFact	0.7500	0.7333	2.650	66.7%
PropagationNet	0.7586	0.7667	2.847	73.3%
SureFact	**0.8136***	**0.8167***	**3.067***	**83.3%***
Improvement	+7.2%	+6.5%	+13.6%	+7.7%

about her/his adjustment according to a 5-point Likert scale. We ensure that for each news article, only one explanation is given to the user, so that the users label a news article only based on this explanation. Different types of explanations are presented in random orders to avoid potential bias.

The results of the user study are given in Table 2. By observing the subgraphs generated by SureFact, human participants can much more accurately decide whether a news article is fake. For example, SureFact improves F1 and Acc by 7.2% and 6.5% compared with the best baseline. This indicates that our subgraphs present more insights for news verification. Moreover, participants tend to be more confident about their labeling results. By comparing labels given by different participants, we find that SureFact also results in better label agreement, which is consistent with the high confidence.

4 RELATED WORKS

The methods of fake news detection can be divided into two major categories: content-based and social-based.

Content-based methods detect fake news by leveraging the textual and/or visual content of the news, or external knowledge about the entities in the news [10, 14, 16, 19, 27, 32, 36, 45, 46, 50]. These methods enable detecting fake news at an early stage, but they lack a mechanism to model important auxiliary data such as the news propagation pattern [48, 49], which may limit their performance.

Social-based methods further integrate useful auxiliary data in the social media for fake news detection, such as user comments about fake news [15, 26, 30, 39], user profiles [41], and user behavior features [3, 19, 38] like "posting" and "re-posting", and user stances [31]. Recently, the news propagation structure in the social media has been shown crucial for improving the detection performance. For example, Ma et al. [28] show the effectiveness of integrating a propagation structure with claims and reply relationships [17]. Yuan et al. [51] improve detection performance by encoding both semantic and structural information. While these methods have shown the effectiveness of propagation network in improving accuracy, how to better enhance explainability is under-explored. Existing works typically explain the prediction by providing a list of important evidence (e.g., posts) or (statistical) features [13, 15, 39, 54]. These method lack detailed information about topological connections, which are crucial for reasoning over graphs [11, 44, 52]. To the best of our knowledge, we propose the first framework for news verification that generates graph-based explanations, which provide connected groups of evidence with convincing details. Our method not only enables a crystal type of explainability, but also improves detection accuracy by increasing the generalization and discrimination power of the detection model.

5 CONCLUSION

We propose SureFact, a subgraph reasoning paradigm for fake news detection. SureFact provides a crystal type of explainability by revealing which subgraphs of the news propagation network are the most important for news verification, and jointly improves the generalization and discrimination power of graph-based detection models by removing task-irrelevant information. Extensive experiments show that our model outperforms the state-of-the-art methods and demonstrate the explainability of our method.

REFERENCES

[1] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *NeurIPS*.

[2] Sergey Brin and Lawrence Page. 1998. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems* 30, 1-7 (1998), 107–117.

[3] Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. 2011. Information credibility on twitter. In *WWW*.

[4] Lu Cheng, Ruocheng Guo, Kai Shu, and Huan Liu. 2021. Causal Understanding of Fake News Dissemination on Social Media. In *KDD*.

[5] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *JMLR* 12, ARTICLE (2011), 2493–2537.

[6] Zhuyun Dai, Chenyan Xiong, Jamie Callan, and Zhiyuan Liu. 2018. Convolutional neural networks for soft-matching n-grams in ad-hoc search. In *WSDM*.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL*.

[8] Y Dou, K Shu, C Xia, PS Yu, and L Sun. 2021. User Preference-aware Fake News Detection. In *SIGIR*.

[9] Yajuan Duan, Zhumin Chen, Furu Wei, Ming Zhou, and Heung Yeung Shum. 2012. Twitter topic summarization by ranking tweets using social influence and content quality. In *COLING*.

[10] Yaqian Dun, Kefei Tu, Chen Chen, Chunyan Hou, and Xiaojie Yuan. 2021. KAN: Knowledge-aware Attention Network for Fake News Detection. In *AAAI*.

[11] Chao Feng, Defu Lian, Xiting Wang, Zheng Liu, Xing Xie, and Enhong Chen. 2022. Reinforcement Routing on Proximity Graph for Efficient Recommendation. *TOIS* (2022).

[12] Dongqi Fu and Jingrui He. 2021. SDG: A Simplified and Dynamic Graph Neural Network. In *SIGIR*.

[13] Jingyue Gao, Xiting Wang, Yasha Wang, Yulan Yan, and Xing Xie. 2021. Learning Groupwise Explanations for Black-Box Models. In *IJCAI*.

[14] Linmei Hu, Tianchi Yang, Luhao Zhang, Wanjun Zhong, Duyu Tang, Chuan Shi, Nan Duan, and Ming Zhou. 2021. Compare to The Knowledge: Graph Neural Fake News Detection with External Knowledge. In *ACL*.

[15] Yiqiao Jin, Xiting Wang, Ruichao Yang, Yizhou Sun, Wei Wang, Hao Liao, and Xing Xie. 2022. Towards Fine-Grained Reasoning for Fake News Detection. In *AAAI*.

[16] Rohit Kumar Kaliyar, Anurag Goswami, and Pratik Narang. 2021. FakeBERT: Fake news detection in social media with a BERT-based deep learning approach. *Multimedia Tools and Applications* 80, 8 (2021), 11765–11788.

[17] Ling Min Serena Khoo, Hai Leong Chieu, Zhong Qian, and Jing Jiang. 2020. Interpretable rumor detection in microblogs by attending to user interactions. In *AAAI*.

[18] Diederik P Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR*.

[19] Sejeong Kwon, Meeyoung Cha, Kyomin Jung, Wei Chen, and Yajun Wang. 2013. Prominent features of rumor propagation in online social media. In *ICDM*.

[20] Stephan Lewandowsky, Ullrich KH Ecker, Colleen M Seifert, Norbert Schwarz, and John Cook. 2012. Misinformation and its correction: Continued influence and successful debiasing. *Psychological science in the public interest* 13, 3 (2012), 106–131.

[21] Jiwei Li, Will Monroe, and Dan Jurafsky. 2016. A simple, fast diverse decoding algorithm for neural generation. *arXiv preprint arXiv:1611.08562* (2016).

[22] Danyang Liu, Jianxun Lian, Zheng Liu, Xiting Wang, Guangzhong Sun, and Xing Xie. 2021. Reinforced Anchor Knowledge Graph Generation for News Recommendation Reasoning. In *KDD*.

[23] Mengchen Liu, Shixia Liu, Xizhou Zhu, Qinying Liao, Furu Wei, and Shimei Pan. 2015. An uncertainty-aware approach for exploratory microblog retrieval. *TVCG* 22, 1 (2015), 250–259.

[24] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* (2019).

[25] Zhenghao Liu, Chenyan Xiong, Maosong Sun, and Zhiyuan Liu. 2020. Fine-grained Fact Verification with Kernel Graph Attention Network. In *ACL*.

[26] Yi-Ju Lu and Cheng-Te Li. 2020. GCAN: Graph-aware Co-Attention Networks for Explainable Fake News Detection on Social Media. In *ACL*.

[27] 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. In *IJCAI*.

[28] Jing Ma, Wei Gao, and Kam-Fai Wong. 2018. Rumor detection on twitter with tree-structured recursive neural networks. ACL.

[29] Sanmit Narvekar, Bei Peng, Matteo Leonetti, Jivko Sinapov, Matthew E Taylor, and Peter Stone. 2020. Curriculum learning for reinforcement learning domains: A framework and survey. *JMLR* (2020).

[30] Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. 2020. BERTweet: A pre-trained language model for English Tweets. In *EMNLP*.

[31] Van-Hoang Nguyen, Kazunari Sugiyama, Preslav Nakov, and Min-Yen Kan. 2020. Fang: Leveraging social context for fake news detection using graph representation. In *CIKM*.

[32] Jeff Z Pan, Siyana Pavlova, Chenxi Li, Ningxi Li, Yangmei Li, and Jinshuo Liu. 2018. Content based fake news detection using knowledge graphs. In *ISWC*. Springer.

[33] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *NeurIPS*.

[34] Kellin Pelrine, Jacob Danovitch, and Reihaneh Rabbany. 2021. The Surprising Performance of Simple Baselines for Misinformation Detection. In *WWW*.

[35] Dean A Pomerleau. 1991. Efficient training of artificial neural networks for autonomous navigation. *Neural computation* 3, 1 (1991), 88–97.

[36] Kashyap Popat, Subhabrata Mukherjee, Andrew Yates, and Gerhard Weikum. 2018. DeClarE: Debunking Fake News and False Claims using Evidence-Aware Deep Learning. In *EMNLP*.

[37] Jon Roozenbeek and Sander van der Linden. 2019. Fake news game confers psychological resistance against online misinformation. *Palgrave Communications* 5, 1 (2019), 1–10.

[38] Natali Ruchansky, Sungyong Seo, and Yan Liu. 2017. Csi: A hybrid deep model for fake news detection. In *CIKM*.

[39] Kai Shu, Limeng Cui, Suhang Wang, Dongwon Lee, and Huan Liu. 2019. defend: Explainable fake news detection. In *KDD*.

[40] Kai Shu, Deepak Mahudeswaran, Suhang Wang, Dongwon Lee, and Huan Liu. 2020. Fakenewsnet: A data repository with news content, social context, and spatiotemporal information for studying fake news on social media. *Big Data* 8, 3 (2020), 171–188.

[41] Kai Shu, Xinyi Zhou, Suhang Wang, Reza Zafarani, and Huan Liu. 2019. The role of user profiles for fake news detection. In *ASNAM*.

[42] Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction*. MIT press.

[43] Soroush Vosoughi, Deb Roy, and Sinan Aral. 2018. The spread of true and false news online. *Science* 359, 6380 (2018), 1146–1151.

[44] Xiting Wang, Kunpeng Liu, Dongjie Wang, Le Wu, Yanjie Fu, and Xing Xie. 2022. Multi-level Recommendation Reasoning over Knowledge Graphs with Reinforcement Learning. In *WWW*. 2098–2108.

[45] Yaqing Wang, Fenglong Ma, Zhiwei Jin, Ye Yuan, Guangxu Xun, Kishlay Jha, Lu Su, and Jing Gao. 2018. Eann: Event adversarial neural networks for multi-modal fake news detection. In *KDD*.

[46] Yaqing Wang, Fenglong Ma, Haoyu Wang, Kishlay Jha, and Jing Gao. 2021. Multimodal Emergent Fake News Detection via Meta Neural Process Networks. In *KDD*.

[47] Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. 2017. End-to-end neural ad-hoc ranking with kernel pooling. In *SIGIR*.

[48] Weizhi Xu, Junfei Wu, Qiang Liu, Shu Wu, and Liang Wang. 2022. Evidence-aware Fake News Detection with Graph Neural Networks. In *WWW*. 2501–2510.

[49] Weizhi Xu, Junfei Wu, Qiang Liu, Shu Wu, and Liang Wang. 2022. Mining Fine-grained Semantics via Graph Neural Networks for Evidence-based Fake News Detection. *WWW*.

[50] Fan Yang, Yang Liu, Xiaohui Yu, and Min Yang. 2012. Automatic detection of rumor on sina weibo. In *KDD*.

[51] Chunyuan Yuan, Qianwen Ma, Wei Zhou, Jizhong Han, and Songlin Hu. 2019. Jointly embedding the local and global relations of heterogeneous graph for rumor detection. In *ICDM*.

[52] Kangzhi Zhao, Xiting Wang, Yuren Zhang, Li Zhao, Zheng Liu, Chunxiao Xing, and Xing Xie. 2020. Leveraging demonstrations for reinforcement recommendation reasoning over knowledge graphs. In *SIGIR*. 239–248.

[53] Cheng Zheng, Bo Zong, Wei Cheng, Dongjin Song, Jingchao Ni, Wenchao Yu, Haifeng Chen, and Wei Wang. 2020. Robust graph representation learning via neural sparsification. In *ICML*.

[54] Xinyi Zhou and Reza Zafarani. 2019. Network-based fake news detection: A pattern-driven approach. In *KDD*.

SUPPLEMENTARY MATERIAL

Appendix I. Implementation Details

Here, we introduce the implementation details.

Construction of the propagation network G. There are four types of nodes in G: claims, posts, users and keywords. Their internal connections are conducted by different relationships. Posts and users are both connected by *reply*/*retweet* relationship. Users are also connected by *reply*/*retweet*. Keywords are connected to each other with *co-occurrence* in a same topic. As for the way the four nodes are connected to each other. Claims are connected with the posts, users, and keywords through edges of the type *similarity*, *discuss*, and *mention*, respectively. The posts are connected with users and keywords by *authorship* and *mention* relationship. And users are also connected with keywords through mention type edges.

Dataset statistics. The detailed statistics of the datasets are shown in Table 3.

Hyperparameter settings. If not mentioned specifically, the number of subgraphs $M = N_{r,1}$ is set to 8. The receptive field sizes $N_{r,2}$ and $N_{r,3}$ at time points 2 and 3 are set to 4 and 2, respectively, so that each subgraph has no more than $4 \times 2 = 8$ nodes. For diversity sibling beam search, the the number of groups is set to 10 and a beam size of 5, with a diverse rate 0.2. For joint optimization, the damping factor in mutural reinforcement ranking is set to 0.85 following [2]. For subgraph modeling, we use the Adam optimizer [18] to update parameters by back-propagation [5] with a learning rate of 5e-5. The maximum length of each post v is set to 130, the same as [15, 22]. We use 10 kernels. One kernel with $\mu_\tau = 1$ and $\sigma_\tau = 1e-3$ is used to model exact matches [6]. The other kernels all have $\sigma_\tau = 0.1$, and their mean μ_τ are evenly distributed within $[-1, 1]$.

Appendix II. Parameter Sensitivity Analysis

Table 4 shows that our method consistently performs better than the two most competitive baselines with different number of kernels, which demonstrates the robustness of SureFact. We also observe that using 10 kernels leads to the best result, while using fewer or more kernels potentially results in underfitting or overfitting.

Appendix III. Proof

Eq. (2) can be proved as follows:

$$\sum_{y\in\{0,1\}} \mathbb{P}(y \mid G)\mathcal{R}^*(y) \tag{28}$$

$$\approx \sum_{y\in\{0,1\}} \sum_{g_{1\sim M}\subset G} \mathbb{P}(y \mid g_{1\sim M})\mathbb{P}(g_{1\sim M} \mid G)\mathcal{R}^*(y) \tag{29}$$

$$= \sum_{\substack{g_{1\sim M}=\{g_1,\dots,g_M\}\\ g_m\subset G}} \mathbb{P}(g_{1\sim M} \mid G) \sum_{y\in\{0,1\}} \mathbb{P}(y \mid g_{1\sim M})\mathcal{R}^*(y) \tag{30}$$

$$= \sum_{g_{1\sim M}\subset G} \mathbb{P}(g_{1\sim M} \mid G)\mathcal{R}(g_{1\sim M}) \tag{31}$$

where the second line is derived based on Eq. (1) and the last line is obtained by using Eq. (3).

Table 3: Statistics of the datasets.

	# True	# Fake	# Total	Avg. $\lvert\mathcal{V}\rvert$	Avg. $\lvert\mathcal{E}\rvert$
PolitiFact	443	372	815	205	4,335
GossipCop	4,219	3,393	7,612	47	763

Table 4: Sensitivity analysis of number of kernels Υ. Underlined and bold text denote best baseline and overall results.

	Politifact		Gossipcop	
	F1	Acc	F1	Acc
BERTweet	0.8658	0.8781	0.8626	<u>0.8478</u>
FinerFact	<u>0.9148</u>	<u>0.9088</u>	<u>0.8657</u>	0.8373
SureFact ($\Upsilon = 5$)	0.9259	0.9403	0.8746	0.8703
SureFact ($\Upsilon = 10$)	**0.9392**	**0.9436**	**0.8811**	**0.8658**
SureFact ($\Upsilon = 15$)	0.9251	0.9310	0.8705	0.8656
SureFact ($\Upsilon = 20$)	0.9213	0.9219	0.8671	0.8608