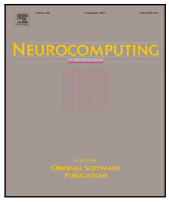
Neurocomputing 604 (2024) 128314

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom



A debiased self-training framework with graph self-supervised pre-training aided for semi-supervised rumor detection

Yuhan Qiao, Chaoqun Cui, Yiying Wang, Caiyan Jia *

School of Computer and Information Technology & Beijing Key Lab of Traffic Data Analysis and Mining, Beijing Jiaotong University, Beijing, 100044, China

ARTICLE INFO

Communicated by M. Gallo

Keywords:
Rumor detection
Self-training
Semi-supervised learning
Self-supervised learning
Confirmation bias
Graph representation

ABSTRACT

Existing rumor detection models have achieved remarkable performance in fully-supervised settings. However, it is time-consuming and labor-intensive to obtain extensive labeled rumor data. To mitigate the reliance on labeled data, semi-supervised learning (SSL), jointly learning from labeled and unlabeled samples, achieves significant performance improvements at low costs. Commonly used self-training methods in SSL, despite their simplicity and efficiency, suffer from the notorious confirmation bias, which can be seen as the accumulation of noise arising from utilization of incorrect pseudo-labels. To deal with the problem, in this study, we propose a debiased self-training framework with graph self-supervised pre-training for semi-supervised rumor detection. First, to enhance the initial model for self-training and reduce the generation of incorrect pseudo-labels in early stages, we leverage the rumor propagation structures of massive unlabeled data for graph self-supervised pre-training. Second, we improve the quality of pseudo-labels by proposing a pseudo-labeling strategy with self-adaptive thresholds, which consists of self-paced global thresholds controlling the overall utilization process of pseudo-labels and local class-specific thresholds attending to the learning status of each class. Extensive experiments on four public benchmarks demonstrate that our method significantly outperforms previous rumor detection baselines in semi-supervised settings, especially when labeled samples are extremely scarce. Notably, we have achieved 96.3% accuracy on Weibo with 500 labels per class and 86.0% accuracy with just 5 labels per class.

1. Introduction

The proliferation of rumors on social media has become a growing concern, as a large amount of false information can spread rapidly, misleading the public and leading to undesirable consequences. Consequently, there has been a significant surge in research dedicated to the automatic detection of rumors [1–11]. Recently, inspired by the expressive capabilities of Graph Neural Networks (GNNs), a considerable amount of research [5–12] have incorporated information about the rumor propagation structures into rumor detection models. These studies carefully design sophisticated models based on the characteristics of rumor dissemination. Undoubtedly, these elaborate designed rumor detection models with deep neural networks have achieved remarkable performance under fully-supervised settings.

However, their successes come at a cost: creating a large-scale annotated dataset of rumors demands a substantial investment in both human and financial resources [13,14]. In practice, annotating a piece of rumor typically requires hiring domain-specific experts to conduct a thorough analysis of claims, along with additional evidence, context, and reports from authoritative sources for debunking [15]. Moreover, rumors usually arise with a newly emergent event and are always blocked by social platforms at early stages. Consequently, obtaining vast amounts of labeled rumor data proves to be time-consuming and labor-intensive. Unlike large language models (LLMs) which have substantial model capacity and are trained with extensive data, existing rumor detection models, characterized by much lower model capacity and dependence on limited labeled data, tend to suffer from over-fitting and exhibit lower robustness.

To alleviate the dependence on labeled data, semi-supervised learning (SSL) is proposed, which utilizes a large set of unlabeled data in combination with a limited set of labeled data to improve model performance. Since unlabeled data is easily accessible, the performance improvements achieved by SSL generally come at low costs. A substantial amount of works on SSL have emerged in the realm of computer vision [16–29]. In the field of rumor detection, as mentioned above, it is hard to obtain a large-scale rumor dataset to train an effective classification model in a fully-supervised manner. SSL with a limited set of labeled data and a large set of unlabeled data is a natural scenario for

* Corresponding author.
E-mail address: cyjia@bjtu.edu.cn (C. Jia).

https://doi.org/10.1016/j.neucom.2024.128314
Received 15 January 2024; Received in revised form 14 June 2024; Accepted 3 August 2024
Available online 8 August 2024

rumor detection tasks. However, there is still a lack of relevant research studying rumor detection in a semi-supervised manner.

The prevailing approaches in SSL can be categorized into two major classes. One branch is self-training [18,19,22,30], also known as pseudo-labeling. Typically, self-training uses the model's predictions on unlabeled data to assign pseudo-labels and then iteratively trains the model with these pseudo-labels. The other branch is consistency regularization [16,17,21,27,29], which enforces the model to produce similar predictions for perturbed unlabeled inputs. In this study, we employ the former method since consistency regularization based approaches typically rely on a rich set of domain-specific data augmentations [30], such as "weak" and "strong" augmentations utilized in numerous studies [16,17,29]. For visual modality data, a variety of data augmentation methods have been extensively researched and developed. However, when it comes to the graph modality data in the context of rumor detection, data augmentation strategies need to be carefully redesigned. Otherwise, the effectiveness of consistency regularization may be limited. In contrast, self-training based approaches inherently do not require data augmentation and can be widely applied across various domains.

Despite its simplicity and efficiency, self-training has a critical flaw in its mechanism, known as the notorious confirmation bias [18]. This bias can be regarded as the accumulation of noise arising from the utilization of incorrect pseudo-labels. Models trained on these erroneous pseudo-labels inherently carry bias, consequently giving rise to more inaccurate predictions. This, in turn, causes the gradual accumulation of noise throughout the iterative self-training process, eventually leading to severe model degradation.

To alleviate confirmation bias during the self-training process, we propose a debiased self-training framework with graph self-supervised pre-training aided for rumor detection. First, we aim to enhance the generalizability and robustness of the initial model of self-training, thereby minimizing the generation of erroneous pseudo-labels in early stages. To achieve this, we utilize massive amounts of unlabeled data for pre-training in an agnostic way, followed by fine-tuning with limited labeled data. Motivated by recent advancements in self-supervised learning [8,25,26,28,31–33], we leverage the rumor propagation structures of unlabeled data for graph self-supervised pre-training. Two typical graph self-supervised learning methods, graph contrastive learning and graph auto-encoder, are employed for pre-training on unlabeled data. In this way, the graph encoder extracts features that are not directly tailored to a specific classification task, leading to better generalization performance and greater resilience to over-fitting. The pre-trained graph encoder, after fine-tuning with labeled data, serves as the initial model for our self-training framework.

We advocate that the key to the success of self-training lies in the strategy for selecting pseudo-labels, as minimizing the utilization of erroneous pseudo-labeled samples helps mitigate confirmation bias. Therefore, we design another debiasing strategy in the phase of pseudo-labels selection. Traditional self-training methods [16,27] typically set a fixed, relatively high threshold as the criterion for pseudo-label selection. To improve the quality of pseudo-labels selected in each iteration of self-training, we devise a pseudo-labeling strategy with self-adaptive thresholds. We first establish a self-paced global threshold based on principles of curriculum learning [22,34], aiming to progressively select unlabeled data from easy to challenging instances. Moreover, we find that a biased model, trained with limited labeled data, tends to predict all unlabeled samples as belonging to one or several specific classes. To attend the learning status of each class, we further introduce a local class-specific threshold based on a class-rebalancing principle. Meanwhile, a distribution alignment module and a fairness regularization term are employed to encourage the model to predict each class equitably. The final threshold is determined by combining both global and local thresholds. Furthermore, the model parameters are reinitialized as the pre-trained parameters after each round of pseudo-label selection, as it helps to prevent the accumulation of bias in subsequent iterations for a model that may have become biased in previous rounds.

We conduct extensive experiments on four public rumor datasets. The results indicate that (1) our method significantly outperforms previously sophisticated and carefully designed rumor detection models in semi-supervised settings, especially in scenarios where the original labeled samples are extremely scarce; (2) compared to traditional self-training methods, our method substantially reduces noise generated during the self-training process, leading to significantly improved performance.

The main contributions of this paper can be summarized as follows:

- To the best of our knowledge, we are the first to explore semi-supervised learning in the context of rumor detection to meet the demands of real-world applications, by presenting a debiased self-training framework which is simple yet effective.
- We utilize both contrastive and generative graph self-supervised methods to pre-train on massive unlabeled data, enhancing the robustness and generalizability of the initial model for self-training, thereby reducing the generation of incorrect pseudo-labels in early stages.
- We improve the quality of pseudo-labels by proposing a pseudo-labeling strategy with self-adaptive thresholds, which consists of the self-paced global thresholds controlling the overall utilization process of pseudo-labels and the local class-specific thresholds attending to the learning status of each class.
- Experimental results on four public benchmark datasets under semi-supervised settings indicate that our model achieves superior performance, even when the original labeled samples are extremely scarce.

2. Related work

2.1. Rumor detection

In recent years, there has been a substantial surge in research efforts dedicated to the automatic detection of rumors. Early studies on automatic rumor detection focus on utilizing handcrafted features based on post contents, user profiles and propagation patterns to train supervised classifiers [35–37]. With the development of deep neural networks, rumor detection methods based on deep learning have achieved significantly better performance. These methods can be categorized into four groups based on the information they leverage: (1) textual contents based methods [1–3], which use the contents from source posts and user reply comments for rumor detection; (2) propagation structure based methods [4–12], which consider the structural information of rumor propagation; (3) multi-source integration methods [38–40], which integrate post content, user profiles and relationships of user-post and user-user pairs as a heterogeneous graph; (4) multi-modal fusion methods [41,42], which combine textual features from posts and visual features from related images for rumor detection.

Among them, propagation structure based methods with GNN have drawn increasing attention due to their superior performance in rumor detection. As one of the pioneer works, Bian et al. [5] use bi-directional graph convolutional networks and a root node feature enhancement strategy to extract rumor features. Wei et al. [6] explore the uncertainty in rumor propagation structures and propose an edge-enhanced Bayesian graph convolutional network to capture robust structural features. Lin et al. [7] present a claim-guided hierarchical graph attention network to fully exploit the information in source posts. These elaborate rumor detection models have achieved remarkable performance under fully-supervised settings. However, when labeled data for training is insufficient, these models tend to over-fit and become less robust. In contrast to these works, in this study, we attempt to avoid designing complex GNN architectures to improve the performance of rumor detection in a fully-supervised manner. Instead, we seek a simple and efficient method for rumor detection in semi-supervised settings.

2.2. Self-training

Self-training, also known as pseudo-labeling, is a key technique widely used in SSL. Typically, its working principle is utilizing the model's predictions on unlabeled data to assign pseudo-labels, followed by iteratively training the model using these pseudo-labeled instances. Lee et al. [19] propose to select the class with the highest predicted probability as pseudo-labels, treating them as if they were true labels. Additionally, pseudo-labels can also be assigned to unlabeled samples based on neighborhood graphs [43]. Despite being simple and efficient, self-training suffers from the problem of confirmation bias [18], also known as noise accumulation. This arises from incorporating incorrect pseudo-labels during subsequent epochs of training. To solve this problem, Rizve et al. [30] propose an uncertainty-aware pseudo-label selection method to enhance the quality of pseudo-labels. Cascante-Bonilla et al. [22] propose an adaptation of curriculum learning to pseudo-labeling, where unlabeled samples are chosen in a self-paced way. Despite substantial research on self-training, it has not been explored in the field of rumor detection yet, although it is a natural learning setting in the real world.

Additionally, consistency regularization stands out as another powerful technique in SSL. It works by enforcing the model to produce similar predictions for perturbed unlabeled inputs. Recently, methods based on consistency regularization [16,17,21,27,29] have gradually outperformed those based on self-training. However, consistency regularization based methods typically necessitate a rich set of domain-specific data augmentations [30]. For visual modality data, such as images, a variety of data augmentation strategies have been extensively researched and developed. Most methods based on consistency regularization [16,17,29] involve performing "weak" augmentation (crop, flip) and "strong" augmentation (Cutout [44], RandAugment [45], CTAugment [24]) on unlabeled images to generate perturbed inputs. However, when dealing with the graph modality data in the context of rumor detection, data augmentation strategies need to be carefully redesigned. Otherwise, the effectiveness of consistency regularization may be limited. In contrast, self-training based methods inherently do not require data augmentation and can be widely applied across various domains.

2.3. Graph self-supervised learning

Self-supervised learning, which leverages pretext tasks formulated solely with unlabeled data, is a general learning framework widely adopted in computer vision and natural language processing. In the realm of graph representation learning, numerous self-supervised methods have been proposed, including graph contrastive and generative approaches. Veličković et al. [46] extend Deep Infomax (DIM) [47] approach to graphs, maximizing the mutual information between global graph embeddings and local node embeddings. Sun et al. [32] focus on maximizing the mutual information between graph-level representations and representations of substructures at various scales. You et al. [48] perform contrastive learning on augmented graph-level representations to enhance the robustness and generalizability of graph encoders. Hou et al. [33] use a masked graph autoencoder to reconstruct masked node features, achieving performance comparable to contrastive methods. Recent studies in computer vision [25,26,28] have demonstrated that applying self-supervised learning to SSL can yield impressive performance. However, concerning graph self-supervised learning methods, there is a lack of research applying them to semi-supervised learning.

3. Problem definition

We formulate the task of semi-supervised rumor detection as learning a model $f_\theta(x)$ from a set of N rumor training samples $\mathcal{D}$. These samples consist of a labeled set $\mathcal{D}_l = \{(x_i, y_i)\}_{i=1}^{N_l}$, with $y_i \in \{0,1\}^C$ being the one-hot label for C classes and an unlabeled set $\mathcal{D}_u = \{x_i\}_{i=1}^{N_u}$ which does not contain label information and $N = N_l + N_u$. Since we adopt the method of self-training, we aim to progressively generate pseudo labels $\tilde{y}$ for the N_u unlabeled samples. Then we reformulate the task as training using $\tilde{\mathcal{D}} = \{(x_i, \tilde{y}_i)\}_{i=1}^{N}$, being $\tilde{y} = y$ for the N_l labeled samples. In our case, x_i is a rumor claim with its all reply posts, being $x_i = \{s_i, r_{i1}, r_{i2}, \ldots, r_{im_i-1}\}$, where s_i is the source post, r_{ij} is the jth reply post and m_i is the number of posts in x_i.

While all reply posts are arranged in sequential order, the entire conversation thread can be constructed as a rumor propagation graph based on reply relationships between the posts. We denote the rumor propagation graph of the ith rumor claim as $G_i = (V_i, E_i)$, where V_i refers to the set of all post nodes with s_i being the root node, and E_i refers to the set of undirected edges corresponding to the reply relationships between nodes in V_i. For example, if r_{i2} is the reply post of r_{i1}, there is an undirected edge between them. We denote the feature matrix and the adjacency matrix of a rumor propagation graph as $X \in \mathbb{R}^{m\times d}$ and $A \in \{0,1\}^{m\times m}$ respectively, where $m = m_i$ represents the number of posts in the rumor propagation graph. For simplicity, we omit the subscript i. And d is the dimension of node features. The model $f_\theta(x)$ is a GNN and θ represents the model parameters. We aim to optimize the model for classifying rumor claims into their corresponding categories as defined by the specific dataset. Typically, there are two kinds of labeling methods:

- Binary labels: rumor and non-rumor, where the task is only to predict whether a claim is a rumor or not [1].
- Four-class labels: non-rumor, true rumor, false rumor and unverified rumor, where finer-grained labels are provided, making rumor detection tasks more challenging [49].

4. Methodology

In this section, we introduce our debiased self-training framework for semi-supervised rumor detection. Fig. 2 presents an overview of our self-training pipeline. In following subsections, we will provide a detailed description of each component of the model.

4.1. Confirmation bias in self-training

As shown in Fig. 1, the traditional self-training method typically starts by training on a limited amount of labeled data. Subsequently, a relatively weak classifier can be obtained, which serves as the initial model for subsequent iterations of self-training. During each iteration, the classifier predicts labels for unlabeled data to generate pseudo-labels. After selecting reliable pseudo-labels based on a pre-defined confidence threshold, these samples are incorporated into the original labeled training set. The classifier is then retrained using this expanded training set. This iterative process repeats for a specified number of iterations or until no more unlabeled data can be reliably labeled.

However, models trained with only a small amount of labeled data inevitably contain bias and tend to over-fit, particularly in previous elaborate rumor detection models [5–7] that are carefully designed based on the characteristics of rumor dissemination. Consequently, a significant portion of the generated pseudo-labels may be incorrect. These erroneous pseudo-labels are then utilized in subsequent iterations of self-training, leading to the accumulation of errors and causing severe model degradation [18,20]. This phenomenon is commonly known as confirmation bias.

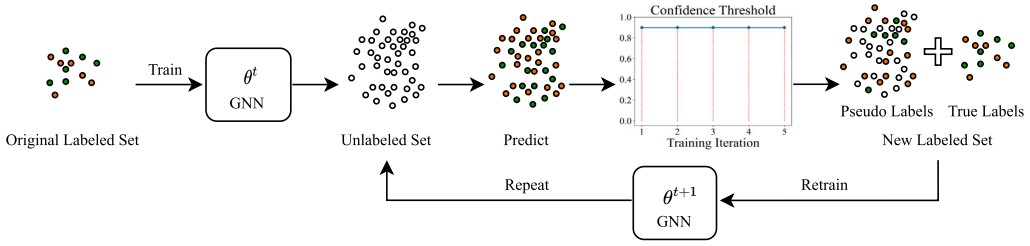


Fig. 1. Illustration of the traditional self-training framework.

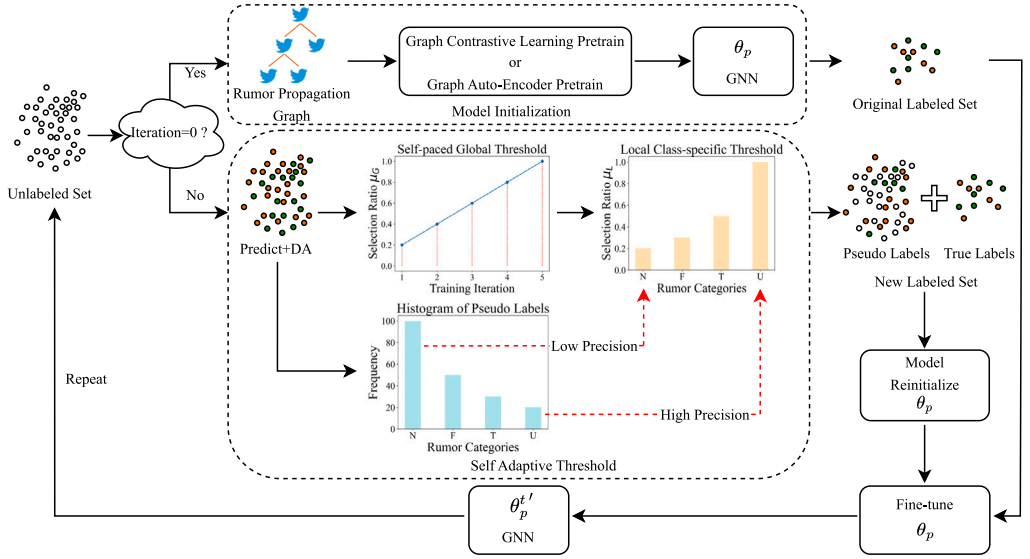


Fig. 2. Illustration of our debiased self-training framework for rumor detection. To initialize the model, we leverage the rumor propagation structures of unlabeled data to pre-train the model using either a graph contrastive learning method or a graph auto-encoder. Subsequently, we fine-tune the pre-trained model using the original labeled data. The model is then employed to predict labels for unlabeled data to generate pseudo-labels. A distribution alignment module is used for calibrating the label guesses. Reliable pseudo-labels are selected based on self-adaptive thresholds and added to the original labeled set. We then reinitialize model parameters as θ_p and fine-tune the model using the enlarged labeled set. This procedure is repeated until no more unlabeled data can be pseudo-labeled.

4.2. Model initialization with self-supervised pre-training

Our goal in this subsection is to reduce noise during initial stages of self-training. Specifically, we aim to enhance the initial model of self-training to achieve stronger performance, thereby generating fewer erroneous pseudo-labels in early stages of self-training.

Inspired by recent advancements in self-supervised learning [8,25,26,28,31–33], we propose leveraging the rumor propagation structures of unlabeled data for graph self-supervised pre-training. Subsequently, we fine-tune the model using the original labeled data and employ this refined model as the initial model for self-training. The rationale behind this approach is that, even though class labels are unavailable for the unlabeled data, their inherent rumor propagation structures contain rich contextual information. By pre-training on a large amount of unlabeled data through pretext tasks, the graph encoder can extract features that are not directly tailored to a specific classification task. This characteristic leads to better generalization performance and greater resilience to over-fitting.

In this study, we adopt two typical graph self-supervised methods: graph contrastive learning and graph auto-encoder, to enhance the generalization performance of the graph encoder. Unlike previous works that involved designing graph data augmentation strategies [8,31,48], such as node dropping and edge removing, the self-supervised methods we use do not necessitate the design of complex data augmentation strategies.

4.2.1. Graph encoder

Unlike previous works designing sophisticated network architectures, we use a vanilla GNN as the graph encoder to obtain representations of rumor propagation graphs. It works by iteratively aggregating information from local node neighborhoods to update node representations. Formally, for a K-layer GNN, the kth layer is

$$a_v^{(k)} = \text{AGGREGATE}^{(k)}\left(\left\{h_u^{(k-1)} : u \in \mathcal{N}(v)\right\}\right), \tag{1}$$

$$h_v^{(k)} = \text{COMBINE}^{(k)}\left(h_v^{(k-1)}, a_v^{(k)}\right), \tag{2}$$

where $h_v^{(k)}$ is the feature vector of node v at the kth layer. $h_v^{(0)}$ is initialized as X_v, and $\mathcal{N}(v)$ is the neighborhood set of v. The choice of $\text{AGGREGATE}^{(k)}(\cdot)$ and $\text{COMBINE}^{(k)}(\cdot)$ depends on the GNN architecture being used. In our framework, there are no specific constraints on the GNN architecture. Namely, it can be GCN [50], GAT [51], or GIN [52], etc.

4.2.2. Graph contrastive self-supervised pre-training

Graph contrastive learning is a prevalent technique in graph self-supervised learning. Most contrastive learning methods require the design of data augmentation strategies [8,10,48]. However, the optimal augmentation strategy may vary across different datasets. To avoid the hassle of data augmentation, we adopt the graph contrastive learning method based on maximizing mutual information between graph-level and node-level representations [32]. We denote the set of

rumor propagation graphs as $\mathbb{G} = \{G_1, G_2, \ldots, G_N\}$.[1] Let ϕ denote the parameters of a K-layer GNN. For a rumor propagation graph $G = (V, E)$, the local patch (node-level) representation $h_\phi^v (v \in V)$ is obtained by concatenating feature vectors at all depths of the GNN into a single feature vector that captures patch information at different scales centered at every node as below.

$$h_\phi^v = \text{CONCAT}\left(\left\{h_v^{(k)}\right\}_{k=1}^{K}\right). \tag{3}$$

Then, the global (graph-level) representation $H_\phi(G)$ can be obtained after applying READOUT operation in the following.

$$H_\phi(G) = \text{READOUT}\left(\left\{h_\phi^v\right\}_{v=1}^{|V|}\right), \tag{4}$$

where $|V|$ is the number of nodes in G. For the calculating of mutual information (MI), we use the Jensen–Shannon MI estimator [53], which is defined on global-local pairs:

$$\begin{aligned} I_{\phi,\psi}\left(h_\phi^v(G); H_\phi(G)\right) =& \mathbb{E}_{\mathbb{P}}\left[-sp\left(-T_\psi\left(h_\phi^v(G), H_\phi(G)\right)\right)\right] \\ &- \mathbb{E}_{\mathbb{P}\times\widetilde{\mathbb{P}}}\left[sp\left(T_\psi\left(h_\phi^v(G'), H_\phi(G)\right)\right)\right], \end{aligned} \tag{5}$$

where $h_\phi^v(G)$ is h_ϕ^v in Eq. (3), $\mathbb{P}$ is the empirical probability distribution of training samples $\mathbb{G}$, the set of rumor propagation graphs on the input space, G is an input sampled from $\mathbb{P}$, G' is a negative instance sampled from $\widetilde{\mathbb{P}} = \mathbb{P}$, $sp(z) = log(1 + e^z)$ is the softplus function, and T_ψ is a discriminator parameterized by a neural network with parameters ψ.

The discriminator takes a (global representation, patch representation) pair as input and determines whether they originate from the same graph. During the training process, for all graph instances in a mini-batch, all possible combinations of global representations and patch representations are generated. Taking any global representation from a batch as an anchor, the patch representations from the same graph are considered as positive samples, forming the first term of Eq. (5). While the patch representations from other graphs are treated as negative samples, forming the second term. To maximize the overall MI, it is desirable to maximize the first term, thereby positive samples are pulled closer to each other; while minimizing the second term, and the negative samples are pushed farther away from each other.

Finally, after averaging the MI across all combinations of global and patch representations over the given dataset $\mathbb{G} = \{G_1, G_2, \ldots, G_N\}$, the contrastive loss can be formulated as

$$\mathcal{L}_{css} = -\frac{1}{N}\sum_{G\in\mathbb{G}}\sum_{v\in V} I_{\phi,\psi}\left(h_\phi^v(G); H_\phi(G)\right). \tag{6}$$

4.2.3. Masked graph auto-encoder pre-training

Graph auto-encoder (GAE) is a type of generative graph self-supervised learning method, which typically consists of an encoder and a decoder. GAE learns graph representations by encoding and reconstructing the input data under the supervision of a reconstruction criterion. We adopt the recently proposed masked graph auto-encoder [33] for self-supervised pre-training.

Formally, for a rumor propagation graph $G = (V, E)$ with $X \in \mathbb{R}^{m\times d}$ and $A \in \{0,1\}^{m\times m}$ being the feature matrix and the adjacency matrix respectively, we randomly sample a subset of nodes $\widetilde{V} \subset V$ and mask each of their features with a token $[\text{MASK}]$, i.e., a learnable vector $x_{[M]} \in \mathbb{R}^d$. Then the node feature $\widetilde{x}_i$ for $v_i \in V$ in the masked feature matrix $\widetilde{X}$ can be defined as

$$\widetilde{x}_i = \begin{cases} x_{[M]} & v_i \in \widetilde{V}, \\ x_i & v_i \notin \widetilde{V}. \end{cases} \tag{7}$$

Subsequently, $\widetilde{X}$ is encoded by the graph encoder f_E in the following equation to obtain the hidden states $H \in \mathbb{R}^{m\times d_h}$.

$$H = f_E\left(A, \widetilde{X}\right). \tag{8}$$

[1] In practice, the propagation graphs of both the labeled set $\mathcal{D}_l$ and the unlabeled set $\mathcal{D}_u$ are used.

To further encourage the graph encoder to learn compressed representations, a re-mask strategy is used to process the latent code H before decoding by replacing H on masked nodes again with another token $[\text{DMASK}]$, i.e., $h_{[M]} \in \mathbb{R}^{d_h}$. Then the re-masked code $\widetilde{h}_i$ in $\widetilde{H} = \text{REMASK}(H)$ can be denoted as

$$\widetilde{h}_i = \begin{cases} h_{[M]} & v_i \in \widetilde{V}, \\ h_i & v_i \notin \widetilde{V}. \end{cases} \tag{9}$$

With the GNN decoder f_D, the re-masked latent code $\widetilde{H}$ is mapped back to reconstruct the input

$$Z = f_D\left(A, \widetilde{H}\right). \tag{10}$$

Finally, given the input feature X and the reconstructed output Z, the scaled cosine error is used as the criterion for computing the reconstruction loss,

$$\mathcal{L}_{sce} = \frac{1}{\left|\widetilde{V}\right|}\sum_{v_i\in\widetilde{V}}\left(1 - \frac{x_i^\top z_i}{\|x_i\|\cdot\|z_i\|}\right)^\eta, \eta \geq 1, \tag{11}$$

where η is a scaling factor adjustable over different datasets. Note that in the inference stage, an input graph is directly fed into the encoder without any masking. Besides, the decoder is only used in the pre-training stage and it will be discarded for inference.

4.2.4. Fine-tuning

In the fine-tuning stage, we initialize parameters of the graph encoder with pre-trained parameters θ_p and fine-tune the model with the labeled set $\mathcal{D}_l$. Formally, after obtaining the node representations h_v by the last layer of GNN, we use a READOUT function to aggregate the node representations and get the final graph representations:

$$h_G = \text{READOUT}\left(\left\{h_v\right\}_{v=1}^{|V|}\right). \tag{12}$$

Then we use a fully connected layer with a softmax activation function to predict the label of the rumor claim as below.

$$\hat{y} = softmax\left(FC\left(h_G\right)\right). \tag{13}$$

We adopt the cross entropy loss of the model predictions $\hat{y}$ and ground truth distributions y over the labeled training set $\mathcal{D}_l$ as below.

$$\mathcal{L}_{sup}\left(\mathcal{D}_l\right) = -\sum_{i=1}^{N_l} y_i log\left(\hat{y}_i\right). \tag{14}$$

In summary, the graph encoder, optimized through self-supervised pre-training and fine-tuning, serves as the initial model for our self-training framework.

4.3. Pseudo-labeling with self-adaptive threshold

We advocate that the key to the success of self-training lies in the strategy for selecting pseudo-labels, as minimizing the utilization of erroneous pseudo-labeled samples helps mitigate confirmation bias. In this subsection, we propose a pseudo-labeling strategy with self-adaptive thresholds to progressively select confident unlabeled samples in each iteration of the self-training process. We first set a self-paced global threshold based on the model's confidence in unlabeled samples, aiming to gradually incorporate unlabeled data from easy to challenging instances. To attend the learning status of each class, we further introduce a local class-specific threshold. The final threshold is then determined by considering both the global and local thresholds.

4.3.1. Self-paced global threshold

Rather than using a fixed, relatively high threshold (e.g., 0.95, as used in [16]), we devise dynamic global thresholds based on the model's confidence in unlabeled samples at each iteration step. Inspired by the idea of curriculum learning [17,22], we set self-paced global thresholds using the percentile of the distribution of maximum probability predictions. In early stages of self-training, only the most confident pseudo-labels are selected. As self-training progresses and the model becomes more stable, we gradually increase the selection ratio to leverage more unlabeled samples. Formally, for the tth self-training iteration step, the global threshold τ_G is defined by

$$\tau_G(t) = \text{Percentile}\left(\mu_G(t)\right), \tag{15}$$

$$\mu_G(t) = \gamma \cdot t \quad , t = 1, 2, \ldots, T, \tag{16}$$

where μ_G is the selection ratio, γ is the incremental stride of it, then $T = 1/\gamma$ denotes the total number of iterations, Percentile $(\cdot)$ is the mapping from the maximum μ_G percentile of the probability distribution for all unlabeled data to the corresponding value of probability predictions. Note that at each selection step, pseudo-labels are selected from the entire unlabeled set, enabling the inclusion or exclusion of previously pseudo-labeled samples in the new training set.

4.3.2. Local class-specific threshold

The global threshold is a strategy for the overall utilization of pseudo-labels across all categories. However, it does not take into account the model's various learning status among different classes. In our experiments, as shown in Fig. 3(a), we find that although the original labeled and unlabeled sets are class-balanced, as self-training progresses, the model tends to predict all unlabeled samples as belonging to one or several specific classes. Without intervention in the pseudo-label selection strategy for each class, pseudo-labels may become even more imbalanced. This phenomenon is similar to findings in research [21], which suggests pseudo-labels are naturally imbalanced due to intrinsic data similarity, even when a model is trained and evaluated on balanced data. We utilize the ground truth labels of unlabeled data here to thoroughly analyze the quality of pseudo-labels predicted as various classes. As shown in Fig. 3(a), (b) and (c), we find that the class predicted most frequently exhibits high recall but low precision, while the class predicted least frequently has high precision but low recall. This implies that pseudo-labels assigned to the least predicted class are more likely to be accurate. Consequently, we design the local class-specific threshold following a class-rebalancing principle: the less frequently a class c is predicted, the more unlabeled samples predicted as class c are selected, and vice versa.

Formally, we calculate the number of pseudo-labeled samples for each class in the tth iteration step as follows.

$$\sigma_t(c) = \sum_{n=1}^{N_u} \mathbb{1}\left(\arg\max\left(p_{m,t}\left(y \mid u_n\right) = c\right)\right), c = 1, 2, \ldots, C, \tag{17}$$

where $\mathbb{1}$ is the indicator function and $p_{m,t}$ is the output probability of the model in the tth iteration. Then we sort $\sigma_t(c)$ in descending order and get $N_1 \geq N_2 \geq \cdots \geq N_C$. Unlabeled samples that are predicted as class c are selected at the ratio of

$$\mu_L(c) = \left(\frac{N_{C+1-c}}{N_1}\right)^{\alpha}, \tag{18}$$

where $\alpha \geq 0$ tunes the sampling ratio. For instance, for a 4-class dataset with the imbalanced pseudo-labels ratio of $\delta = \frac{N_1}{N_4} = 10$, the sample ratio of the most minority class is $\mu_L(4) = \left(\frac{N_{4+1-4}}{N_1}\right)^{\alpha} = 1$ and the sample ratio of the most majority class is $\mu_L(1) = \left(\frac{N_{4+1-1}}{N_1}\right)^{\alpha} = 0.1^{\alpha}$. When $\alpha = 0$, $\mu_L(c) = 1$ for all classes, then all pseudo-labels are kept. Note that we select the most confident samples when selecting in each class.

The final threshold τ_t is obtained after integrating the self-paced global threshold and the local class-specific threshold

$$\tau_t(c) = \text{Percentile}\left(\mu_t(c)\right), \tag{19}$$

$$\mu_t(c) = \mu_G(t) \cdot \mu_L(c), \tag{20}$$

where $\mu_t(c)$ is the selection ratio for class c in the tth iteration step. Finally, the unsupervised loss in the tth iteration is

$$\mathcal{L}_u = \frac{1}{N_u}\sum_{n=1}^{N_u} \mathbb{1}\left(\max\left(q_n\right) \geq \tau_t\left(\hat{q}_n\right)\right) \cdot \mathcal{H}\left(\hat{q}_n, q_n\right), \tag{21}$$

where $q_n = p_{m,t}\left(y \mid u_n\right)$, $\hat{q}_n = \arg\max\left(q_n\right)$ is the "hard" pseudo-label and $\mathcal{H}(\cdot,\cdot)$ refers to cross-entropy loss.

4.3.3. Distribution alignment

To further improve the quality of pseudo-labels, we additionally integrate a distribution alignment (DA) module. DA was first proposed in [24], aiming to align the model's predictive distribution on unlabeled samples with the labeled training set's class distribution $p(y)$. It works by scaling the model's prediction $q_n = p_m\left(y \mid u_n\right)$ for an unlabeled sample u_n by the ratio $p(y)/\tilde{p}(y)$, where $\tilde{p}(y)$ is the moving average of the model's predictions on unlabeled samples. We slightly modify the ratio by adding a tuning knob β, where $\beta \geq 0$, to control the strength of DA. Finally, we re-normalize the scaled value to form a valid probability distribution as below.

$$\tilde{q}_n = \text{Normalize}\left(q_n\left(\frac{p(y)}{\tilde{p}(y)}\right)^{\beta}\right). \tag{22}$$

$\tilde{q}_n$ will be used as the label guess for u_n instead of q_n. Note that DA is only used in each pseudo-labeling step to calibrate the label guess. Once pseudo-labeling is done, DA will not be applied during the subsequent training.

4.4. Training strategy

After acquiring pseudo-labels for unlabeled data, we devise a training strategy to efficiently leverage the combination of original labeled and pseudo-labeled data. Specifically, we ensure that all original labeled data are evenly distributed across each mini-batch. Consequently, each mini-batch includes a minimum number of accurate labels, which helps to mitigate the confirmation bias. Similarly, the supervised loss for labeled data at the tth iteration is

$$\mathcal{L}_s = \frac{1}{N_l}\sum_{n=1}^{N_l} \mathcal{H}\left(y_n, p_{m,t}\left(y \mid x_n\right)\right). \tag{23}$$

Additionally, we reinitialize the model parameters as the pre-trained parameters θ_p after each iteration of self-training, as it can prevent the accumulation of bias in subsequent iterations for a model that may have been biased in previous rounds.

To address the issue that a poorly calibrated model at an early stage may blindly predict all unlabeled samples as one or several specific classes, a widely-used fairness regularization term [18] is adopted. The term showed below encourages the model to predict each class with the same frequency.

$$\mathcal{L}_f = \sum_{c=1}^{C} p_c \log\left(\frac{p_c}{\bar{h}_c}\right), \tag{24}$$

where p_c is the prior probability distribution for class c, $\bar{h}_c$ is the mean softmax probability of the model for class c across all samples in the dataset. Since all the datasets we use are class-balanced, we assume a uniform distribution $p_c = 1/C$ for the prior probabilities.

Then, the overall objective for our self-training framework is formulated as

$$\mathcal{L} = \mathcal{L}_s + \lambda_u \mathcal{L}_u + \lambda_f \mathcal{L}_f, \tag{25}$$

where λ_u and λ_f are hyper-parameters of loss weights for $\mathcal{L}_u$ and $\mathcal{L}_f$, respectively.

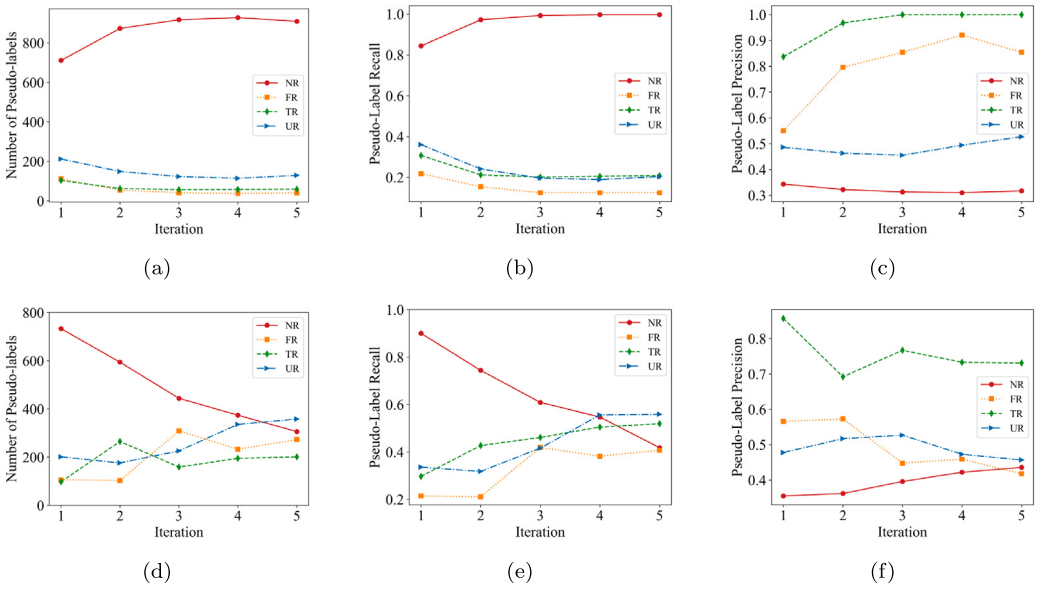


Fig. 3. Experimental results on Twitter15 with 40 labeled samples used per class. (a), (b) and (c) represent the curves depicting the quantity, recall, and precision of pseudo-labels for each class without the local class-specific threshold module during iterations. In contrast, (d), (e), and (f) illustrate the curves using our self-training method. With the help of the local class-specific threshold module, the quantity of pseudo-labels for each class becomes more balanced, enabling the model to predict each class more equitably.

Table 1
Statistics of datasets.

Statistics	Weibo	DRWeibo	Twitter15	Twitter16
Language	zh	zh	en	en
# claims	4664	6037	1490	818
# non-rumors	2351	3185	374	205
# false rumors	2313	2852	370	205
# true rumors	–	–	372	207
# unverified rumors	–	–	374	201
Avg. # comments	803.5	61.8	50.2	49.1

5. Experiments

5.1. Datasets

We evaluate our model on four publicly available real-world benchmark datasets, Weibo [1], DRWeibo [54], Twitter15 [49] and Twitter16 [49]. Weibo and DRWeibo are Chinese rumor datasets with binary labels, including non-rumor (NR) and rumor (R). Twitter15 and Twitter16 are English rumor datasets which contain four finer-grained labels, including non-rumor (NR), false rumor (FR), true rumor (TR) and unverified rumor (UR). In this study, we focus on class-balanced semi-supervised rumor detection, so the datasets we use have a similar number of samples for each class. Detailed statistical information for all datasets can be found in Table 1.

5.2. Experimental setup

We compare our proposed model with the following baselines:

- GIN [52] : A commonly used GNN architecture for graph classification tasks. GIN generalizes the Weisfeiler-Lehman (WL) test and achieves maximum discriminative power among GNNs. Note that we take undirected rumor propagation graphs as the model input for GIN.
- BiGCN [5] : A GCN-based rumor detection model which uses two GCN to encode top-down and bottom-up rumor propagation graphs to learn high-level representations.
- ClaHi-GAT [7] : A GAT-based rumor detection model which uses hierarchical attention at the post-level and the event-level to capture multi-level rumor indicative features.
- GIN-GCLP [32] : A GIN encoder whose parameters are Pre-trained by Graph Contrastive Learning as mentioned in Section 4.2.2 and fine-tuned by the original labeled set. GIN-GCLP can be seen as the initial model for our proposed self-training framework RDST-GCLP.
- GIN-GAEP [33] : A GIN encoder whose parameters are Pre-trained by the Graph Auto-Encoder as mentioned in Section 4.2.3 and fine-tuned by the original labeled set. We use GIN as the graph encoder and decoder for the auto-encoder architecture. GIN-GAEP can be seen as the initial model for our proposed self-training framework RDST-GAEP.
- CL [22] : A representative self-training method leveraging curriculum learning for semi-supervised learning. We re-implement this method for the task of rumor detection, with GIN used as the backbone graph encoder. Furthermore, we additionally incorporate the distribution alignment module described in Section 4.3.3, denoted as CL-DA.

Since our method adopts Graph Contrastive Learning Pre-training and Graph Auto-Encoder Pre-training to initialize the proposed Self-Training framework for Rumor Detection, we name our model as RDST-GCLP and RDST-GAEP, respectively. To ensure reproducibility, we have made the source code of RDST publicly available.[2]

For the split of datasets, Weibo and DRWeibo are split into training/validation/testing sets with a ratio of 6:2:2. Twitter15 and Twitter16 datasets contain relatively small amounts of data, hence we split these two datasets into training/testing sets with a ratio of 8:2 and conduct 5-fold cross-validation, in line with previous works [5–7]. Each split has the same distribution of classes, which means all splits are class-balanced. To conduct semi-supervised experiments, the training set is further divided into a labeled subset and an unlabeled subset with various ratios. Regarding the size of datasets, the number of labeled samples per class is chosen from {5, 10, 20, 40, 100, 200, 500} for Weibo and DRWeibo, {10, 20, 40, 80, 100} for Twitter15 and {10, 20, 40, 80} for Twitter16. The remaining samples of each dataset are kept as the unlabeled set. The maximum value chosen ensures that the number of labeled samples is less than the number of unlabeled samples. Labeled samples are randomly selected, and to reduce bias

[2] https://github.com/qyhan97/RDST.

Table 2
Semi-supervised rumor detection results on Weibo dataset.

	Acc.							
# Label	5	10	20	40	100	200	500	All
GIN	0.691	0.708	0.746	0.791	0.874	0.907	0.940	0.945
BiGCN	0.631	0.653	0.762	0.832	0.879	0.909	0.945	0.948
ClaHi-GAT	0.665	0.705	0.773	0.823	0.887	0.914	0.947	0.953
GIN-GCLP	0.766	0.773	0.831	0.856	0.912	0.927	0.946	0.962
GIN-GAEP	0.759	0.823	0.854	0.861	0.900	0.924	0.957	**0.966**
CL	0.687	0.712	0.764	0.836	0.896	0.927	0.949	–
CL-DA	0.691	0.714	0.768	0.848	0.898	0.932	0.951	–
RDST-GCLP	**0.860**	**0.868**	**0.895**	**0.917**	**0.937**	0.945	0.960	–
RDST-GAEP	0.824	0.853	0.878	0.909	0.930	**0.951**	**0.963**	–

Table 3
Semi-supervised rumor detection results on DRWeibo dataset.

	Acc.							
# Label	5	10	20	40	100	200	500	All
GIN	0.557	0.586	0.667	0.713	0.739	0.775	0.819	0.868
BiGCN	0.553	0.609	0.650	0.701	0.754	0.797	0.836	0.885
ClaHi-GAT	0.568	0.614	0.654	0.705	0.751	0.780	0.841	**0.888**
GIN-GCLP	0.562	0.607	0.721	0.749	0.769	0.798	0.834	0.876
GIN-GAEP	0.559	0.592	0.685	0.726	0.747	0.792	0.839	0.882
CL	0.553	0.591	0.714	0.740	0.777	0.792	0.827	–
CL-DA	0.573	0.614	0.719	0.742	0.778	0.796	0.830	–
RDST-GCLP	**0.653**	**0.689**	**0.746**	**0.771**	**0.801**	0.825	0.852	–
RDST-GAEP	0.626	0.65	0.708	0.751	0.787	**0.838**	**0.866**	–

Table 4
Semi-supervised rumor detection results on Twitter15 dataset.

	Acc.					
# Label	10	20	40	80	100	All
GIN	0.445	0.524	0.625	0.737	0.757	0.848
BiGCN	0.495	0.577	0.678	0.753	0.791	0.880
ClaHi-GAT	0.507	0.585	0.676	0.760	0.785	0.875
GIN-GCLP	0.464	0.556	0.663	0.756	0.788	0.865
GIN-GAEP	0.501	0.588	0.699	0.778	0.809	**0.887**
CL	0.456	0.549	0.665	0.766	0.808	–
CL-DA	0.479	0.562	0.676	0.771	0.812	–
RDST-GCLP	0.569	0.653	0.740	0.793	0.827	–
RDST-GAEP	**0.603**	**0.665**	**0.771**	**0.825**	**0.842**	–

Table 5
Semi-supervised rumor detection results on Twitter16 dataset.

	Acc.				
# Label	10	20	40	80	All
GIN	0.510	0.594	0.688	0.779	0.868
BiGCN	0.561	0.656	0.737	0.816	0.879
ClaHi-GAT	0.565	0.675	0.765	0.820	0.887
GIN-GCLP	0.549	0.632	0.720	0.803	0.883
GIN-GAEP	0.566	0.641	0.737	0.825	**0.892**
CL	0.537	0.625	0.769	0.812	–
CL-DA	0.549	0.629	0.771	0.815	–
RDST-GCLP	0.635	0.753	0.805	0.839	–
RDST-GAEP	**0.670**	**0.774**	**0.819**	**0.857**	–

introduced by sampling, we randomly sample five times and obtain five different labeled/unlabeled sets for each data scale.

We use 5000-dimensional word frequency vectors as initial node features for Twitter15 & Twitter16, and 200-dimensional word2vec [55] embeddings as initial node features for Weibo & DRWeibo. The layer of GNN is set to 2. We use mean-pooling as the READOUT function for all the GNNs. The dimension of hidden vectors are set to 64 for Twitter15 & Twitter16 and 128 for Weibo & DRWeibo. The incremental stride of pseudo-label selection ratio γ is set to 0.1 for Weibo & DRWeibo and 0.2 for Twitter15 & Twitter16. The class-rebalancing strength α is set to 0.5, 0.25, 1.0, 1.0 for Weibo, DRWeibo, Twitter15 and Twitter16, respectively. The DA strength β is set to 0.5. The loss weights λ_u and λ_f are set to 0.8 and 0.4, respectively. The batch sizes for Twitter15, Twitter16, Weibo and DRWeibo are set to 256, 128, 32, 32, respectively. We train the model via back-propagation with Adam optimizer [56]. The learning rate and the weight decay are 0.0005 and 0.0001, respectively. The optimal set of hyperparameters is chosen by evaluating performance on the fold-0 set of Twitter15 & Twitter16 and the validation set of Weibo & DRWeibo. We implement all models on PyTorch[3] and all baseline methods are re-implemented.

5.3. Semi-supervised rumor detection performance

The experimental results for semi-supervised rumor detection on Weibo, DRWeibo, Twitter15 and Twitter16 datasets are reported in Tables 2, 3, 4 and 5 respectively. The bold numbers represent the best performance. In line with previous SSL studies [16,17,22,29], we report the classification accuracy under various labeled data scales. "# Label" in the tables indicates the number of labeled data used for each category, and "All" represents that the entire training set are labeled data (fully-supervised).[4]

Based on Tables 2–5, we have following findings.

(1) Although elaborate rumor detection models, carefully designed based on the characteristics of rumor dissemination, perform very well when abundant labeled samples are available, there is a significant risk of over-fitting when labeled data is extremely scarce. For instance, when each class has only 5 or 10 labeled samples, these sophisticated models may perform even worse than simple baseline models. This is particularly obvious on Weibo dataset, given only 5 labeled samples per class, BiGCN and ClaHi-GAT exhibit accuracies that are 6.0% and 2.6% lower than the vanilla GIN.

(2) GIN-GCLP and GIN-GAEP, utilizing unlabeled data for graph self-supervised pre-training and fine-tuned by labeled data, consistently outperform the baseline GIN across all labeled data scales. When compared to the sophisticated models BiGCN and ClaHi-GAT, GIN-GCLP and GIN-GAEP achieve either superior or competitive performance. Promising results can be observed on Weibo dataset. In semi-supervised settings, GIN-GAEP demonstrates accuracy improvements ranging from 1.0% to 17.0% compared to BiGCN and ClaHi-GAT across various labeled data scales. Similarly, in fully-supervised settings, GIN-GAEP achieves an accuracy 1.8% higher than BiGCN and 1.3% higher than ClaHi-GAT. These results highlight the increased resilience to over-fitting achieved by our proposed graph self-supervised pre-training methods, thereby enhancing the generalization performance of the graph encoder. Consequently, GIN-GCLP and GIN-GAEP can be considered as suitable initial models for self-training to alleviate confirmation bias in early stages.

(3) The representative self-training method CL, benefiting from its efficient pseudo-labeling strategy, consistently outperforms its backbone model GIN across the majority of labeled data scales. However, with only 5 labeled samples per class available, CL performs worse than GIN on Weibo and DRWeibo datasets, as shown in Tables 2 and 3. This validates our earlier claim that models trained with extremely limited labeled data inevitably contain bias, and as self-training progresses, the models gradually accumulates errors, leading to degradation in performance. With the assistance of the distribution alignment module, CL-DA encourages the class distribution of unlabeled data predicted by the model to match the actual class distribution. While there is a slight improvement in classification accuracy, a noticeable gap remains when compared to the proposed methods RDST-GCLP and RDST-GAEP.

[3] https://pytorch.org/.

[4] The # Label for "All" column are 1399, 1811, 298, 163, respectively for Tables 2–5.

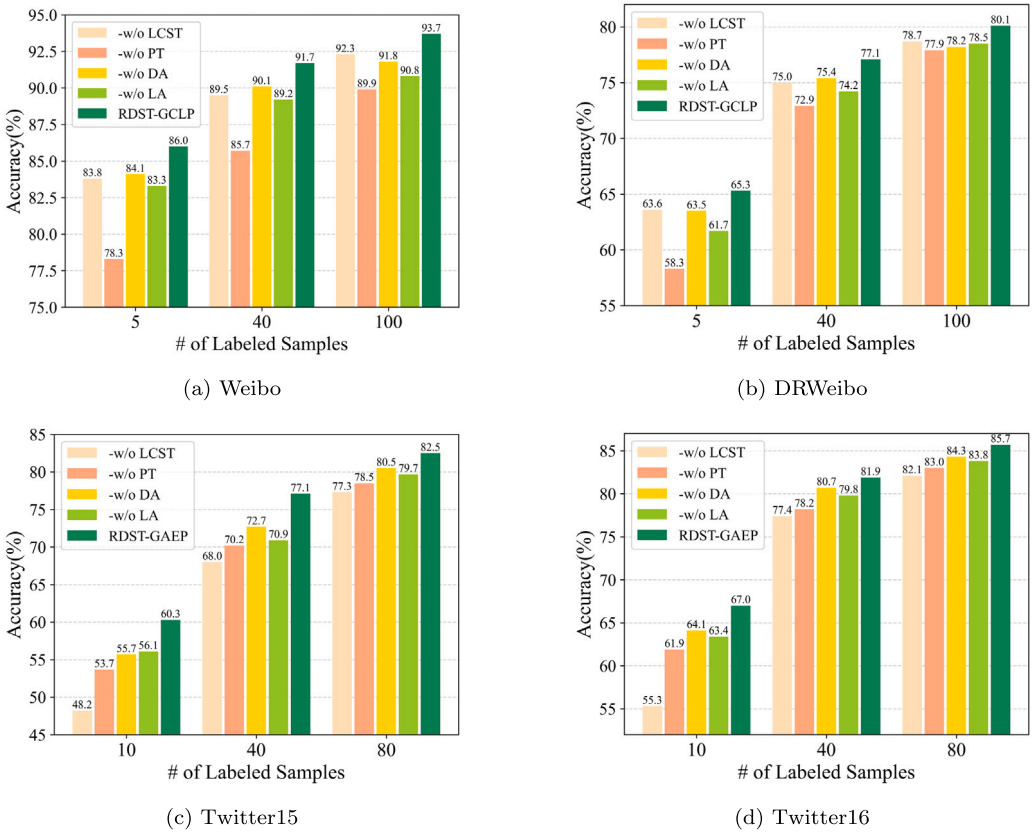


Fig. 4. Ablation study results of RDST on four datasets.

(4) Our proposed RDST-GCLP and RDST-GAEP significantly outperform all baselines across all labeled data scales in the four datasets. Surprisingly, our method demonstrates remarkably promising results even in scenarios with extremely limited labeled data. For instance, when given 5 labeled samples per class, RDST-GCLP outperforms CL by 17.3% and 10.0% in accuracy on Weibo and DRWeibo datasets, respectively. Similarly, with 10 labeled samples per class, RDST-GAEP surpasses CL by 14.7% and 13.3% in accuracy on Twitter15 and Twitter16 datasets, respectively. Furthermore, at the largest labeled data scale, the performance of RDST-GAEP is either close or superior to that of baseline models in fully-supervised settings. Above results demonstrate that our proposed model, with the help of self-supervised pre-training initialization and self-adaptive threshold modules, significantly mitigates confirmation bias, thereby enhancing the efficiency of self-training.

5.4. Ablation study

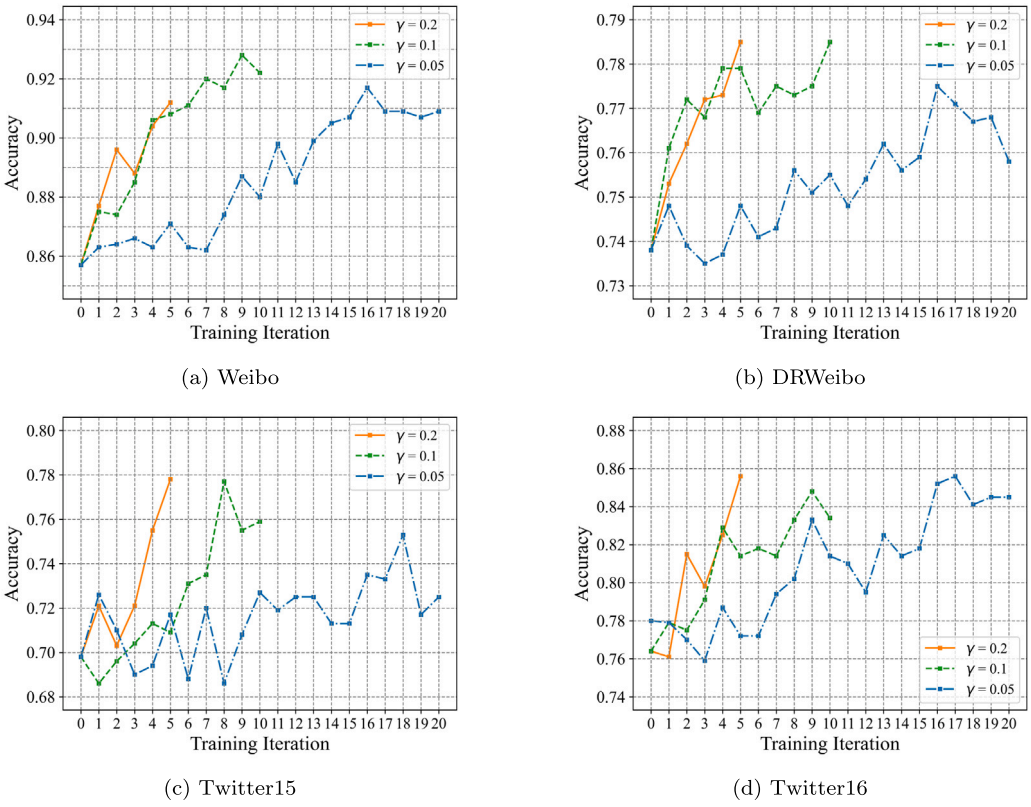
To better understand the effectiveness of each component in RDST, we conduct ablation studies by excluding certain crucial components. We select three different scales of labeled samples across four datasets, including $\{5, 40, 100\}$ for Weibo & DRWeibo and $\{10, 40, 80\}$ for Twitter15 & Twitter16. We compare our model with the following variants, (1) -w/o LCST where we neglect the local class-specific threshold in the pseudo-labeling stage; (2) -w/o PT where we omit the graph self-supervised pre-training for GNN model initialization (the model is initialized randomly and trained directly using labeled data); (3) -w/o DA where we discard the distribution alignment module; (4) -w/o LA where the original labeled samples are randomly allocated across each mini-batch of the new training set, instead of being averagely allocated.

As shown in Fig. 4, generally, each component contributes to the improvement of RDST. Specifically, -w/o LCST exhibits a significant decrease in accuracy on Twitter15 and Twitter16 datasets. This could be attributed to the fact that Twitter15 & Twitter16 are four-class datasets, containing more rumor categories compared to binary-class datasets Weibo & DRWeibo. Due to intrinsic data similarity, a model trained on insufficient labeled data may exhibit relatively weak classification capability, making inter-class confusion more likely to occur. Therefore, the LCST module, which sets class-specific thresholds based on the various learning status of each category during the self-training process, selects more accurate pseudo-labels for each class, and is particularly crucial for the Twitter datasets. However, for Weibo and DRWeibo datasets, which only contain two rumor categories, the LCST module is not the most crucial factor influencing the entire framework. Instead, the graph self-supervised pre-training for initialization is more important, as -w/o PT exhibits the worst performance among all variants on Weibo & DRWeibo datasets. We believe the potential reason is that Weibo & DRWeibo datasets have a greater quantity of samples compared to Twitter15 & Twitter16 datasets, as shown in Table 1. Consequently, there are more rumor propagation graphs available for graph self-supervised pre-training, thereby enhancing the generalization of the graph encoder to a greater extent. Both -w/o DA and -w/o LA lead to a decrease in accuracy, demonstrating their effectiveness in calibrating and leveraging pseudo-labels, respectively.

Effect of Self-paced Global Threshold. To thoroughly investigate the effect of self-paced global threshold, we adopt several fixed confidence thresholds {0.8, 0.9, 0.95} in self-training for comparison. The number of iteration steps is set to be consistent with those used for self-paced thresholds. Table 6 presents the number of pseudo-labels selected (#PL for short) and the testing accuracy at each iteration step. The results indicate that using a relatively high fixed threshold leads to unstable performance during the self-training process, and there is a significant gap observed between the final accuracy of fixed thresholds and our self-paced threshold. In contrast, the pseudo-labeling strategy with self-paced thresholds progressively exploits unlabeled samples

Table 6
Performance comparison among fixed and self-paced thresholds.

	Threshold							
	0.8		0.9		0.95		Self-paced	
Iteration step	# PL	Accuracy	# PL	Accuracy	# PL	Accuracy	# PL	Accuracy
Iteration-0	–	0.853	–	0.853	–	0.853	–	0.853
Iteration-1	1978	0.863	1464	0.866	1159	0.864	255	0.865
Iteration-2	1835	0.872	1096	0.865	1540	0.848	504	0.864
Iteration-3	2038	0.870	432	0.852	463	0.854	747	0.868
Iteration-4	2212	0.874	707	0.865	1406	0.851	1079	0.872
Iteration-5	1760	0.886	585	0.863	1560	0.844	1317	0.874
Iteration-6	1597	0.882	1342	0.859	572	0.836	1473	0.873
Iteration-7	2265	0.889	826	0.868	1530	0.849	1813	0.883
Iteration-8	2008	0.891	1215	0.854	1594	0.857	2081	0.899
Iteration-9	2194	0.887	1047	0.850	1527	0.835	2355	0.907
Iteration-10	2041	0.887	1378	0.861	1524	0.842	2370	0.910



(a) Weibo (b) DRWeibo

(c) Twitter15 (d) Twitter16

Fig. 5. The effect of various incremental strides γ.

from easy to hard, simultaneously considering the quality and quantity of pseudo-labels for selection. The accuracy steadily improves as self-training progresses.

The incremental stride of global threshold γ determines the quantity of confident pseudo-labels selected during each self-training iteration. To choose an optimal incremental stride γ that facilitates the rapid convergence of self-training while ensuring the quality of pseudo-labels, we experiment with different incremental strides {0.05, 0.1, 0.2}. We conduct experiments on all four datasets with 40 labeled samples per class. Fig. 5 shows the accuracy changes per round of self-training iterations under various γ. The results indicate that the best performance is achieved when γ is set to 0.1, 0.1, 0.2, and 0.2 for Weibo, DRWeibo, Twitter15 and Twitter16, respectively. Reducing γ does not lead to better performance, instead, it leads to more unstable performance and increased computational time, especially noticeable on DRWeibo and Twitter15 datasets.

Effect of Class-rebalancing Strength α. Fig. 6(a) shows the accuracy score against α. When $\alpha = 0$, the sampling ratio is set to 1 for all classes, rendering the local class-specific thresholds completely disabled. Without class-rebalancing, the model exhibits significant bias towards certain classes, resulting in the worst performance across four datasets. The best performance is achieved when α is set to 0.50, 0.25, 1.00 and 1.00 for Weibo, DRWeibo, Twitter15 and Twitter16, respectively. We speculate that the reason for the larger optimal α in Twitter15 and Twitter16 is that these two datasets contain more rumor categories compared to Weibo and DRWeibo, making it easier for inter-class confusion to occur. On the other hand, an excessively large α makes the class-rebalancing sampling too strong, resulting in imbalance towards the reversed direction and causing a degradation in performance.

Effect of Distribution Alignment Strength β. Fig. 6(b) shows the accuracy score against β. When $\beta = 0$, the DA module is disabled, resulting in consistently poor performance across all four datasets. This is attributed to the fact that, in early stages of self-training, a biased model trained with limited labeled data may blindly predict the majority of unlabeled samples as a single class, leading to generation

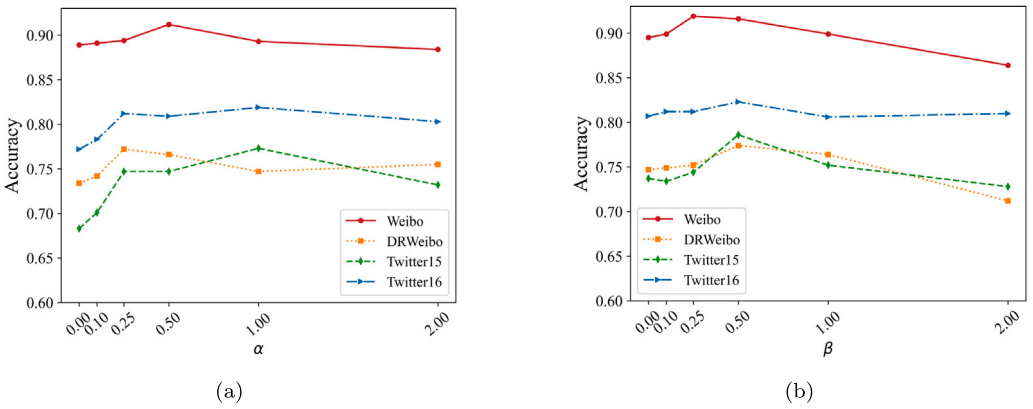


Fig. 6. The effect of the class-rebalancing strength α (left) and the distribution alignment strength β (right).

Table 7
The effect of trade-off hyper-parameters λ_u and λ_f. Bold values indicate the best performance. Underlined values indicate the results obtained with the configuration we use.

# Label	5					40				
λ_u/λ_f	0.1	0.4	0.8	1.0	2.0	0.1	0.4	0.8	1.0	2.0
0.1	0.824	0.795	0.829	0.833	0.857	0.890	0.901	0.891	0.903	0.895
0.4	0.825	0.832	0.861	0.841	0.832	0.890	0.907	0.892	0.902	0.880
0.8	0.812	0.860	**0.869**	0.840	0.842	0.892	**0.916**	0.907	0.885	0.901
1.0	0.842	0.833	0.817	0.857	0.863	0.893	0.895	0.902	0.901	0.896
2.0	0.832	0.778	0.814	0.807	0.840	0.889	0.890	0.881	0.892	0.898

of numerous incorrect pseudo-labels. The DA module addresses this issue by aligning the predicted class distribution of the model for unlabeled samples with the class distribution of the labeled training set, calibrating the previous label guess. As β increases, the performance of self-training gradually improves. The best performance is achieved when β is 0.25, 0.50, 0.50 and 0.50 for Weibo, DRWeibo, Twitter15 and Twitter16, respectively. As β continues to increase, the performance does not show a corresponding improvement. We speculate that this may be attributed to the pseudo-label distribution becoming overly balanced, with more samples incorrectly predicted as the minority (hardly biased) class, thereby reducing the precision of the minority class. This contradicts our original intention to exploit them for generating better pseudo-labels.

Effect of Trade-off Hyper-parameters λ_u and λ_f. Table 7 shows the accuracy under various combinations of trade-off hyper-parameters λ_u and λ_f. Our configuration ($\lambda_u = 0.8$ and $\lambda_f = 0.4$) achieves good performance and generalizes well. An excessively small or large value of λ_u may cause the model to underestimate or overestimate the pseudo-labels, leading to degraded performance. Similarly, maintaining an appropriate value for λ_f ensures that the model predicts each class fairly.

5.5. Feature visualization

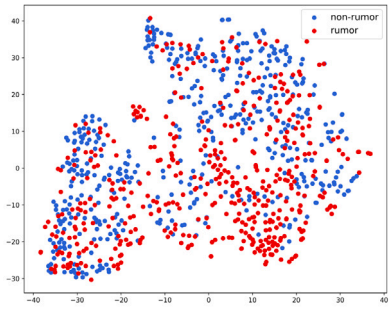
To provide a more intuitive portrayal of feature representations learned by the proposed method across different classes, we conduct a visualization study by using T-SNE. We choose two datasets, Weibo and Twitter15, and employ the following methods to extract graph feature representations of all test data instances. Subsequently, we visualize these representations alongside their original labels.

For Weibo dataset, with only 5 labeled data available per class, the comparison of graph representations learned by ClaHi-GAT, GIN-GCLP, CL and RDST-GCLP is shown in Fig. 7. It can be observed that when labeled data is extremely scarce, ClaHi-GAT struggles to learn discriminative features across various classes, leading to a lack of clear cluster formation. Conversely, GIN-GCLP, following extensive utilization of unlabeled data for graph self-supervised pre-training, enhances the generalization capability of the graph encoder, yielding distinct features that segregate into two discernible clusters. Therefore, adopting GIN-GCLP as the initial model for the proposed self-training framework RDST-GCLP is deemed reasonable. The typical self-training framework CL, lacking modules designed for mitigating bias, although it obtains more compact feature representations, features from two classes are heavily mixed and intertwined. However, RDST-GCLP, benefiting from graph self-supervised pre-training and employing a pseudo-labeling strategy with self-adaptive thresholds, alleviates the confirmation bias during self-training and learns more compact and distinguishable features.

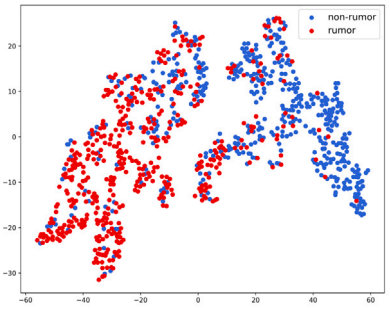
For Twitter15 dataset, with 40 labeled data available per class, we extract graph representations using ClaHi-GAT, GIN-GAEP, CL and RDST-GAEP. As shown in Fig. 8, we can find similar results where ClaHi-GAT fails to generate discriminative graph feature representations. Conversely, through pre-training with the masked graph auto-encoder, the features extracted by GIN-GAEP can form several localized clusters. However, the boundaries of each cluster remain somewhat ambiguous. For the typical self-training method CL, although it forms four distinct clusters, a notable bias towards the NR class is evident within the model. Specifically, the cluster of the NR class contains a substantial number of samples that actually belong to other three classes, a phenomenon consistent with our observations in Section 4.3.2. Consequently, the model tends to predict these samples as the NR class, resulting in high recall but low precision for the NR class. Conversely, other three classes exhibit higher precision. Therefore, it is reasonable to follow the principle of selecting more pseudo-labels from minority classes and fewer from majority classes when designing local class-specific thresholds. Benefiting from pre-training with the masked graph auto-encoder and particularly the local class-specific threshold, RDST-GAEP alleviates the bias of the model to a certain extent. Notably, the presence of samples from other classes in the cluster of NR has been significantly reduced.

6. Conclusions and future work

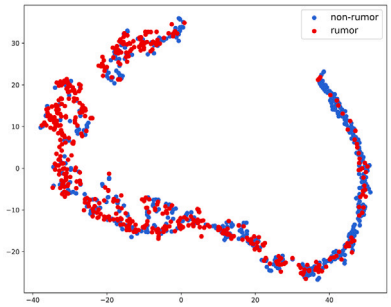
In this work, we conduct the first study on semi-supervised learning for rumor detection and propose a debiased self-training framework. To alleviate the confirmation bias and reduce noise in the initial stage of self-training, we leverage the rumor propagation structure of massive



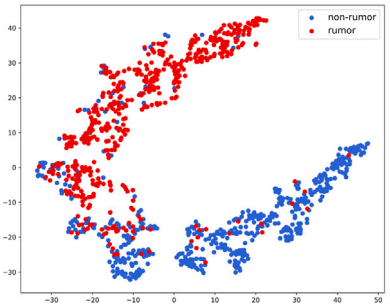
(a) ClaHi-GAT



(b) GIN-GCLP

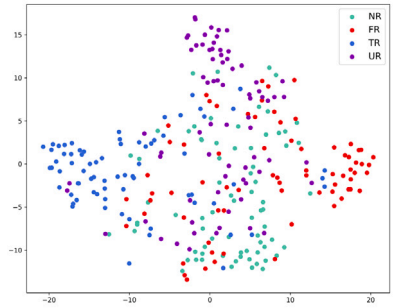


(c) CL

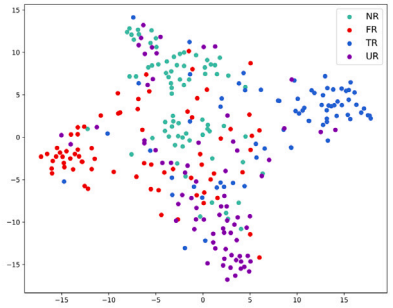


(d) RDST-GCLP

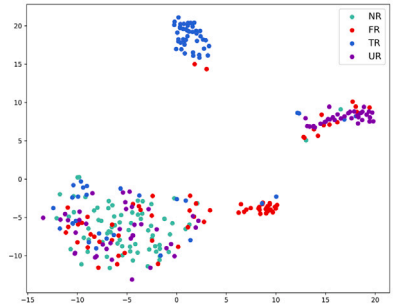
Fig. 7. Visualization of feature representations on Weibo dataset.



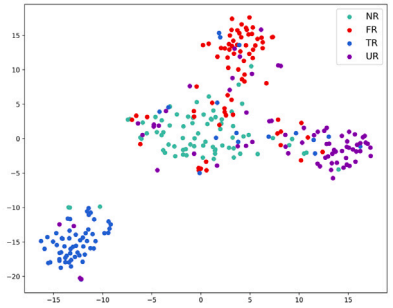
(a) ClaHi-GAT



(b) GIN-GAEP



(c) CL



(d) RDST-GAEP

Fig. 8. Visualization of feature representations on Twitter15 dataset.

unlabeled samples to perform graph self-supervised pre-training. In this way, the generalization and robustness of the initial graph encoder are enhanced. To improve the quality of pseudo-labels selected in each iteration of self-training, we propose a pseudo-labeling strategy with self-adaptive thresholds, which consists of self-paced global thresholds controlling the overall utilization process of pseudo-labels and local class-specific thresholds attending the learning status of each class. Through the implementation of a suite of semi-supervised learning techniques, including distribution alignment, model re-initialization, and fairness regularization, we further alleviate the confirmation bias and improve the performance of self-training. Results on four public benchmarks show that (1) our method significantly outperforms previous rumor detection baselines in semi-supervised settings, especially in scenarios where original labeled samples are extremely scarce; (2) compared to traditional self-training methods, our method substantially reduces noise generated during the self-training process, leading to significantly improved performance.

In the future, the following directions deserve to be concerned. (1) The class distribution of rumor data in real-world situations may not always be balanced. It is a significant challenge to investigate semi-supervised rumor detection with class-imbalanced data. (2) Although methods based on consistency regularization have dominated in semi-supervised learning, the "strong" and "weak" augmentation strategies for rumor data have not been explored yet. It is valuable to develop effective augmentation strategies for rumor data and explore consistency regularization-based methods for semi-supervised rumor detection.

CRediT authorship contribution statement

Yuhan Qiao: Writing – original draft, Methodology, Investigation, Conceptualization. **Chaoqun Cui:** Methodology, Data curation. **Yiying Wang:** Methodology, Investigation. **Caiyan Jia:** Writing – review & editing, Methodology, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

I have shared the link to my code.

Acknowledgments

This work is supported in part by the National Key R&D Program of China (2018AAA0100302) and the National Natural Science Foundation of China (61876016).

References

[1] J. Ma, W. Gao, P. Mitra, S. Kwon, B.J. Jansen, K. Wong, M. Cha, Detecting rumors from microblogs with recurrent neural networks, in: IJCAI, 2016, pp. 3818–3824.
[2] F. Yu, Q. Liu, S. Wu, L. Wang, T. Tan, A convolutional approach for misinformation identification, in: IJCAI, 2017, pp. 3901–3907.
[3] Y. Liu, Y.B. Wu, Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks, in: AAAI, 2018, pp. 354–361.
[4] J. Ma, W. Gao, K. Wong, Rumor detection on Twitter with tree-structured recursive neural networks, in: ACL, 2018, pp. 1980–1989.
[5] T. Bian, X. Xiao, T. Xu, P. Zhao, W. Huang, Y. Rong, J. Huang, Rumor detection on social media with bi-directional graph convolutional networks, in: AAAI, 2020, pp. 549–556.
[6] L. Wei, D. Hu, W. Zhou, Z. Yue, S. Hu, Towards propagation uncertainty: Edge-enhanced Bayesian graph convolutional networks for rumor detection, in: ACL/IJCNLP, 2021, pp. 3845–3854.
[7] H. Lin, J. Ma, M. Cheng, Z. Yang, L. Chen, G. Chen, Rumor detection on Twitter with claim-guided hierarchical graph attention networks, in: EMNLP, 2021, pp. 10035–10047.
[8] Z. He, C. Li, F. Zhou, Y. Yang, Rumor detection on social media with event augmentations, in: SIGIR, 2021, pp. 2020–2024.
[9] L. Tian, X. Zhang, J.H. Lau, DUCK: Rumour detection on social media by modelling user and comment propagation networks, in: NAACL, 2022, pp. 4939–4949.
[10] T. Sun, Z. Qian, S. Dong, P. Li, Q. Zhu, Rumor detection on social media with graph adversarial contrastive learning, in: WWW, 2022, pp. 2789–2797.
[11] Y. Gao, X. Wang, X. He, H. Feng, Y. Zhang, Rumor detection with self-supervised learning on texts and social graph, Front. Comput. Sci. 17 (4) (2023) 174611.
[12] P. Zhang, H. Ran, C. Jia, X. Li, X. Han, A lightweight propagation path aggregating network with neural topic model for rumor detection, Neurocomputing 458 (2021) 468–477.
[13] K. Sharma, F. Qian, H. Jiang, N. Ruchansky, M. Zhang, Y. Liu, Combating fake news: A survey on identification and mitigation techniques, ACM Trans. Intell. Syst. Technol. 10 (3) (2019) 21:1–21:42.
[14] X. Zhou, R. Zafarani, A survey of fake news: Fundamental theories, detection methods, and opportunities, ACM Comput. Surv. 53 (5) (2021) 109:1–109:40.
[15] K. Shu, A. Sliva, S. Wang, J. Tang, H. Liu, Fake news detection on social media: A data mining perspective, SIGKDD Explor. 19 (1) (2017) 22–36.
[16] K. Sohn, D. Berthelot, N. Carlini, Z. Zhang, H. Zhang, C. Raffel, E.D. Cubuk, A. Kurakin, C. Li, FixMatch: Simplifying semi-supervised learning with consistency and confidence, in: NeurIPS, 2020, pp. 596–608.
[17] B. Zhang, Y. Wang, W. Hou, H. Wu, J. Wang, M. Okumura, T. Shinozaki, FlexMatch: Boosting semi-supervised learning with curriculum pseudo labeling, in: NeurIPS, 2021, pp. 18408–18419.
[18] E. Arazo, D. Ortego, P. Albert, N.E. O'Connor, K. McGuinness, Pseudo-labeling and confirmation bias in deep semi-supervised learning, in: IJCNN, 2020, pp. 1–8.
[19] D.-H. Lee, et al., Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks, in: Workshop on Challenges in Representation Learning, ICML, Vol. 2, 2013, p. 896.
[20] B. Chen, J. Jiang, X. Wang, P. Wan, J. Wang, M. Long, Debiased self-training for semi-supervised learning, in: NeurIPS, 2022, pp. 32424–32437.
[21] X. Wang, Z. Wu, L. Lian, S.X. Yu, Debiased learning from naturally imbalanced pseudo-labels, in: CVPR, 2022, pp. 14627–14637.
[22] P. Cascante-Bonilla, F. Tan, Y. Qi, V. Ordonez, Curriculum labeling: Revisiting pseudo-labeling for semi-supervised learning, in: AAAI, 2021, pp. 6912–6920.
[23] D. Berthelot, N. Carlini, I.J. Goodfellow, N. Papernot, A. Oliver, C. Raffel, MixMatch: A holistic approach to semi-supervised learning, in: NeurIPS, 2019, pp. 5050–5060.
[24] D. Berthelot, N. Carlini, E.D. Cubuk, A. Kurakin, K. Sohn, H. Zhang, C. Raffel, ReMixMatch: Semi-supervised learning with distribution alignment and augmentation anchoring, in: ICLR, 2020.
[25] L. Beyer, X. Zhai, A. Oliver, A. Kolesnikov, S4L: Self-supervised semi-supervised learning, in: ICCV, 2019, pp. 1476–1485.
[26] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, G.E. Hinton, Big self-supervised models are strong semi-supervised learners, in: NeurIPS, 2020, pp. 22243–22255.
[27] Q. Xie, Z. Dai, E.H. Hovy, T. Luong, Q. Le, Unsupervised data augmentation for consistency training, in: NeurIPS, 2020, pp. 6256–6268.
[28] Y. Zhang, X. Zhang, J. Li, R.C. Qiu, H. Xu, Q. Tian, Semi-supervised contrastive learning with similarity Co-calibration, IEEE Trans. Multimedia 25 (2023) 1749–1759.
[29] Y. Wang, H. Chen, Q. Heng, W. Hou, Y. Fan, Z. Wu, J. Wang, M. Savvides, T. Shinozaki, B. Raj, B. Schiele, X. Xie, FreeMatch: Self-adaptive thresholding for semi-supervised learning, in: ICLR, 2023.
[30] M.N. Rizve, K. Duarte, Y.S. Rawat, M. Shah, In defense of pseudo-labeling: An uncertainty-aware pseudo-label selection framework for semi-supervised learning, in: ICLR, 2021.
[31] J. Zeng, P. Xie, Contrastive self-supervised learning for graph classification, in: AAAI, 2021, pp. 10824–10832.
[32] F. Sun, J. Hoffmann, V. Verma, J. Tang, InfoGraph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization, in: ICLR, 2020.
[33] Z. Hou, X. Liu, Y. Cen, Y. Dong, H. Yang, C. Wang, J. Tang, GraphMAE: Self-supervised masked graph autoencoders, in: KDD, 2022, pp. 594–604.
[34] Y. Bengio, J. Louradour, R. Collobert, J. Weston, Curriculum learning, in: ICML, 2009, pp. 41–48.
[35] C. Castillo, M. Mendoza, B. Poblete, Information credibility on twitter, in: WWW, 2011, pp. 675–684.
[36] S. Kwon, M. Cha, K. Jung, W. Chen, Y. Wang, Prominent features of rumor propagation in online social media, in: ICDM, IEEE Computer Society, 2013, pp. 1103–1108.

[37] J. Ma, W. Gao, Z. Wei, Y. Lu, K. Wong, Detect rumors using time series of social context information on microblogging websites, in: CIKM, 2015, pp. 1751–1754.
[38] Q. Huang, J. Yu, J. Wu, B. Wang, Heterogeneous graph attention networks for early detection of rumors on Twitter, in: IJCNN, 2020, pp. 1–8.
[39] C. Yuan, Q. Ma, W. Zhou, J. Han, S. Hu, Jointly embedding the local and global relations of heterogeneous graph for rumor detection, in: ICDM, IEEE, 2019, pp. 796–805.
[40] H. Ran, C. Jia, P. Zhang, X. Li, MGAT-ESM: Multi-channel graph attention neural network with event-sharing module for rumor detection, Inform. Sci. 592 (2022) 402–416.
[41] Z. Jin, J. Cao, H. Guo, Y. Zhang, J. Luo, Multimodal fusion with recurrent neural networks for rumor detection on microblogs, in: ACM MM, 2017, pp. 795–816.
[42] Y. Wang, F. Ma, Z. Jin, Y. Yuan, G. Xun, K. Jha, L. Su, J. Gao, EANN: Event adversarial neural networks for multi-modal fake news detection, in: KDD, 2018, pp. 849–857.
[43] A. Iscen, G. Tolias, Y. Avrithis, O. Chum, Label propagation for deep semi-supervised learning, in: CVPR, 2019, pp. 5070–5079.
[44] T. DeVries, G.W. Taylor, Improved regularization of convolutional neural networks with cutout, 2017, arXiv:1708.04552.
[45] E.D. Cubuk, B. Zoph, J. Shlens, Q.V. Le, RandAugment: Practical automated data augmentation with a reduced search space, 2019, arXiv:1909.13719.
[46] P. Velickovic, W. Fedus, W.L. Hamilton, P. Liò, Y. Bengio, R.D. Hjelm, Deep graph infomax, in: ICLR, 2019.
[47] R.D. Hjelm, A. Fedorov, S. Lavoie-Marchildon, K. Grewal, P. Bachman, A. Trischler, Y. Bengio, Learning deep representations by mutual information estimation and maximization, in: ICLR, 2019.
[48] Y. You, T. Chen, Y. Sui, T. Chen, Z. Wang, Y. Shen, Graph contrastive learning with augmentations, in: NeurIPS, 2020, pp. 5812–5823.
[49] J. Ma, W. Gao, K. Wong, Detect rumors in microblog posts using propagation structure via kernel learning, in: R. Barzilay, M. Kan (Eds.), ACL, 2017, pp. 708–717.
[50] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, in: ICLR, 2017.
[51] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, Y. Bengio, Graph attention networks, in: ICLR, 2018.
[52] K. Xu, W. Hu, J. Leskovec, S. Jegelka, How powerful are graph neural networks? in: ICLR, 2019.
[53] S. Nowozin, B. Cseke, R. Tomioka, F-GAN: Training generative neural samplers using variational divergence minimization, in: NeurIPS, 2016, pp. 271–279.
[54] C. Cui, C. Jia, Propagation tree is not deep: Adaptive graph contrastive learning approach for rumor detection, in: AAAI, 2024, pp. 73–81.
[55] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, in: ICLR, 2013.
[56] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, in: ICLR, 2015.

Yuhan Qiao, born in 1997, post graduate. His main research interests include rumor detection, graph representation learning and semi-supervised learning.

Chaoqun Cui, born in 1999, post graduate. His main research interests include natural language processing, rumor detection and graph representation learning.

Yiying Wang, born in 1997, post graduate. Her main research interests include natural language processing, rumor detection and semi-supervised learning.

Caiyan Jia, born in 1976, professor. Her main research interests include data mining, complex network analysis and natural language processing.