Applied Soft Computing 128 (2022) 109438

Contents lists available at ScienceDirect

Applied Soft Computing

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

# Metapath and syntax-aware heterogeneous subgraph neural networks for spam review detection

Zhiqiang Zhang, Yuhang Dong, Haiyan Wu *, Haiyu Song, Shengchun Deng, Yanhong Chen

*School of Information Management and Artificial Intelligence, Zhejiang University of Finance and Economics, Hangzhou, China*

ARTICLE INFO

*Article history:*
Received 25 January 2022
Received in revised form 15 July 2022
Accepted 27 July 2022
Available online 2 August 2022

*Keywords:*
Spam detection
Syntactic parse
Semantic parse
Metapath
Heterogeneous graph

ABSTRACT

Spam Review Detection is a subclass of text classification that aims to distinguish genuine reviews from spam reviews (e.g., irrelevant reviews, deceptive reviews, machine-generated reviews, and non-review messages). Previous studies have focused on review text analysis, abnormal behavior detection, and intrinsic relationship identification. However, these methods ignore different fraudulent camouflages and writing styles. In this paper, we instead design a **Spam** detection model **M**etapath-based **S**ubgraph **A**ggregated **N**eural **N**etwork (**Spam-MSANN**) integrating three metapath-based subgraphs (i.e. User-Item Subgraph, Review Subgraph, and User-Review-Item Subgraph) and syntactic information to enhance the relevant representation of the review information with subgraph aggregation operations. Experimental results on benchmarking datasets (i.e. YelpCHI and Amazon) demonstrate that our Spam-MSANN model significantly improves the state-of-the-art models. Specifically, Spam-MSANN outperforms 11 of the 12 advanced benchmark models on these two datasets, which further manifests that the fusion of different metapath-based subgraphs and syntactic information is adequate for the spam review detection task.



## 1. Introduction

With the development of online social networks and e-commerce, there have been more and more spam reviews. Simultaneously, the explosive growth of online users has led to the existence of a large amount of user review information on all major platforms, which not only provides a reference for users but also provides guidance for merchants to optimize their products or services. Spam reviews prevent users and merchants from genuinely grasping the use of products or services and thus making wrong judgments and decisions [1,2]. Therefore, spam review detection is essential.

On spam review detection task, many scholars were tried to employ different methods and have achieved some remarkable results. The good reviews are so profitable for merchants, more and more merchants tend to hire people to post spam reviews to improve the ratings of their products and stores, and a survey [3] showed that the percentage of spam on Yelp had risen from 6% in 2006 to 20% in 2013. Spam reviews mislead consumer decisions and damage the consumer experience in a harmful way, so detecting spam reviews on online social networks and review platforms has become an urgent yet arduous task. Jindal et al. first introduced the concept of spam review detection and classified spam reviews into three categories: deceptive reviews, irrelevant reviews, and non-review messages [4]. Irrelevant reviews and non-review messages (e.g., advertisements) are easily identified by detection systems and individuals, while deceptive reviews are difficult to identify [4]. The difficulty of detecting deceptive reviews lies in the fact that review contents are mostly short texts [5]. It is difficult to acquire practical features to better separate real from deceptive reviews. The slow development of detection techniques is also because there are not many annotated datasets and it is hard to do manual annotation [6,7]. In general, existing work has two aspects of shortcomings: for users and merchants, it is challenging to identify widespread behavior that is camouflaged, and for platforms, there are technical limitations. Specifically, there are four shortcomings in two aspects that make deceptive reviews challenging to identify.

(1) Review text camouflage. Most reviews of products are generally copy-pasted, pieced together, or written by a particular person, rather than generated by natural language processing technology, and current work is mainly for detecting review text [8–10], but it is difficult to identify simply from the review meaning and sentence structure and requires a lot of domain knowledge and advanced recognition algorithms to help detect. (2) Behavior camouflage. To counter some existing spam user filtering mechanisms, spam review publishers also apply various camouflage to normal users, such as posting long reviews, reviews with pictures, or merchants buying normal users' accounts

* Corresponding author.
*E-mail addresses:* zqzhang@zufe.edu.cn (Z. Zhang), yhdong@zufe.edu.cn (Y. Dong), wuhy2020@zufe.edu.cn (H. Wu), hysong@zufe.edu.cn (H. Song), dengsc@zufe.edu.cn (S. Deng), cyhong@zufe.edu.cn (Y. Chen).

https://doi.org/10.1016/j.asoc.2022.109438
1568-4946/

to post spam reviews. The existing work is mainly divided into detecting user and item features [11,12] and review features [13, 14], but these efforts are only partially effective. (3) Relation-based co-camouflage. In many cases, review text camouflage and behavior camouflage exist simultaneously, and the detector can put only the action of a user reviewing an item into the relationship context to detect the intrinsic relationship between them [15–17]. So [18] established the first User-Review-Item heterogeneous graph to explore the intrinsic relationships between data points in relation-based co-camouflage detection. (4) Technical limitations: The level of development of computer science and artificial intelligence limits the growth of spam review detection. Previous studies have only detected the camouflage methods mentioned above in a different perspective or performed a simple combination, but they cannot be applied to all the other camouflage methods that are widespread today [19]. With the development of graph neural networks(GNNs) [20,21], GNNs are trained by aggregating neighborhood information and learning node representations, saving data annotation costs by semi-supervised training. Recent research has demonstrated that graph neural networks incorporating review embeddings as edge or node representation better represent user behavior in purchasing items in spam detection and outperform structure and ensemble learning [15,16,19]. Previous work introduced reinforcement learning into GNNs [22,23], and later work used reinforcement learning to filter suspicious nodes in spam review detection [16, 17]. However, applying these GNN-based detection methods to all different camouflage methods is impossible.

To address these shortcomings, we put forward a metapath and syntax-aware heterogeneous subgraph neural network to detect spam reviews. Specifically, we establish a Spam-Graph incorporating syntactic information and build the subgraph representation and node aggregation methods for each of the three subgraphs describing different fraud camouflages: (1) For review camouflage, we construct a User-Item subgraph and learn the node embeddings through metapath to detect the abnormal behavior of users or items. (2) For behavioral camouflage, we create a review subgraph and detect abnormal reviews through syntactic and semantic encoders. (3) For co-camouflage, we build a User-Review-Item subgraph, then we employ the inter-metapath aggregation and the intra-metapath aggregation to detect abnormal relationships. These subgraph representation and node aggregation methods mine the user's writing style syntactic similarity, item's review semantic similarity, behavioral consistency, and the intrinsic relationship consistency of users reviewing items. Finally, subgraph aggregation is applied to detect fraudulent camouflage and thus improve spam review detection. The main contributions of this work are summarized as follows:

- We construct different metapaths and metapath-based subgraphs with subgraph aggregation operations to detect the camouflage information for different camouflage methods.
- We design a **Spam** detection model **M**etapath-based **S**ubgraph **A**ggregated **N**eural **N**etwork (**Spam-MSANN**) aggregating three metapath-based subgraphs (i.e. User-Item Subgraph, Review Subgraph, and User-Review-Item Subgraph) and syntactic information is proposed to enhance the relevant representation of the review information for the spam review detection task.
- Experimental results on the benchmark datasets (YelpCHI and Amazon) manifest that our Spam-MSANN method significantly outperforms 11 out of 12 state-of-the-art methods on detecting spam reviews.

## 2. Related work

In the past few years, it has become important to discover spam review, and more and more researchers are working on this problem. [4] first introduced the concept of spam review detection and early work used statistical or machine learning methods to model and detect review features, user behavior, and item behavior [4,11,12,24]. [25] created a hotel review dataset through a crowdsourcing platform, which became the first publicly available spam review detection dataset. Ahmadi et al. studied the importance of expert systems in industry [26], while [15] used an expert system to mark spam reviews. Feature such as user behavioral features and review features [4,27,28] are an effective class of influencing factors, and a part of spam users are successfully detected by their review frequency [11], rating distribution [11], and review burstiness [27]. [29] used a hybrid Deep-Reinforcement model to extract features and select the best users. Besides, most of the studies focused on the features of review texts [13,14]. They considered that spam reviews might be somewhat different from benign reviews in terms of manual features of reviews or semantic features such as sentiment [30], and therefore extracted features by POS n-grams or bags of words [12]. In addition to manual features, a large number of methods are used to learn features automatically. For example, generating features automatically for detection by movie spam review identification algorithms using generative adversarial networks [31] or employing dynamic feature selection techniques [32] to detect spam users on Twitter. [33] presents a density-based outlier spam detection approach.

However, manual review features cannot represent the meaning of a review text. With the development of natural language processing techniques, the deeper semantic meanings of review text are continuously uncovered by text pre-training and sentence embedding methods. Text representation methods can adequately express the meaning of review text, so in addition to review features, existing work uses text representations as a way of spam review detection. [10] used Word2Vec [34] in SMS text message filtering and [15] used TextCNN [35] to learn sentence embeddings for the representation of reviews. [9] presented a hierarchical attention network for text classification. [36] introduced Transformer and later works such as BERT [37] and XLNET [38] presented a series of pre-training models based on Transformer and proved to be effective in spam review detection [39,40]. However, these detection methods do not consider the review's syntactic structure.

• **Review Syntactic Parse.** In addition, research [41,42] shows that syntactic analysis can learn information that semantic does not have and has been used to better effect in many domains [43]. [42,44] proves that syntactic information can enhance textual representation. [45,46] demonstrates the effectiveness of syntactic parsing in authorship identification, and [43] demonstrates that syntactic structure increases user identification in recommendation systems. Therefore, in this paper, we incorporate the syntactic structure of Spam-Graph and prove its effectiveness.

The contribution of manual features and text representations to spam reviews is not separate [47]. More and more advanced relationship-based detection models use artificial neural networks [31] to integrate features and text representations to learn the intrinsic relationship between the features and review text, achieving better performance in spam review detection.

• **Metapath and Heterogeneous Graph.** Unlike the simple combination of features and text representations above, the act of a user having reviewed an item is well suited to being represented by a heterogeneous graph, where the graph structure is advantageous for expressing the relationship between users and items. So [18] established the first User-Review-Item heterogeneous graph to explore the intrinsic relationships between data

points in relation-based co-camouflage detection. Multiple types of nodes and edges can exist in a heterogeneous graph, allowing different nodes to have different dimensional features or attributes, so heterogeneous graph is a more representative of realistic and widespread network structure. In heterogeneous graphs, most studies obtain graph semantic information through different kinds of metapaths [48,49], where a metapath is an ordered sequence containing node and edge types [50]. Setting different metapaths in heterogeneous graphs can capture different semantic information under different tasks, which also demonstrates the superior generalization ability of heterogeneous graphs and metapath-based subgraphs for different downstream tasks [48,49,51]. For example, previous work [48,52] used metapaths and heterogeneous graphs for node classification and link prediction, and Liang et al. applied them to mobile APP recommendation [53].

- **GNN-based Spam Detection.** In recent years, semi-supervised graph neural networks (GNNs) have achieved great success in learning graph structure and node information [20,21]. GAS [15] used heterogeneous and homogeneous graphs to capture local and global contextual reviews and detect spam reviews through a convolutional model of heterogeneous graphs based on metapaths. GraphConsis [54] came up with the idea of a multi-relational heterogeneous graph, where the nodes are review features and the edges are the relationships between them. This helped solve the problems of inconsistent feature, relation, and context for fraudulent users on multi-relational graphs. R-GCN [55] and SemiGNN [56] proposed further improvements to relational heterogeneous graph neural networks and were used in financial fraud detection [56] and online app review system fraud detection [19] with good results. CARE-GNN [16] and RioGNN [17] proposed a relational heterogeneous graph neural network to address fraudster feature camouflage and relationship camouflage by discovering information-rich neighboring nodes with a label-aware similarity-based metric and disclosing the optimal number of neighbors and aggregating neighbors by reinforcement learning. RLC-GNN [57] used Residual Layer to improve the performance of CARE-GNN, and [58] improved spam review detection for learning on non-homophilous graphs.

In Spam-MSANN, we design heterogeneous graphs to model that users have reviewed items and detect different camouflage methods using different metapaths and metapath-based subgraphs, respectively. Our paper built a Spam-Graph using and enhancing the detection of features, text, and relationships. We apply a User-Item subgraph and metapath contextual relationships to detect abnormal user and item behavior. We establish a review subgraph to detect abnormal writing syntactic styles of users as well as semantic relevance abnormalities of reviews. We build a User-Review-Item subgraph to detect co-camouflages by anomalous relationships. Due to different camouflage methods' simultaneous existence, we jointly detect these camouflages by subgraph aggregation. Experimental results illustrate that our method is better at detecting different camouflage behaviors at the same time and is more thorough than what has been done before.

## 3. Preliminaries and problem definition

In this section, we elaborate on some of the important definitions used in the paper.

**Definition 1. Pre-trained Semantic Parsing** aims to translate natural language into their formal meaning representations [59,60]. We adopt advanced text pre-trained models to convert natural language to vectors that contain sufficient semantic information [37].

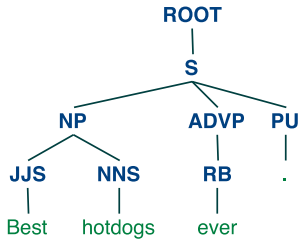


**Fig. 1.** An example of a syntactic tree.

**Table 1**
Review features.

| Symbol | Definition |
|---|---|
| $\mathbf{X}^R_{\text{RES}}$ | Ratio of exclamation sentences containing'!'[24] |
| $\mathbf{X}^R_{\text{PC}}$ | Percentage of capital letters [24] |
| $\mathbf{X}^R_{\text{Length}}$ | Review Length [24] |
| $\mathbf{X}^R_{\text{PCW}}$ | Percentage of ALL-capitals words [24] |
| $\mathbf{X}^R_{\text{PP1}}$ | Ratio of 1st person pronouns [24] |
| $\mathbf{X}^R_{\text{OW}}$ | Ratio of objective words(By sentiWordNet [64])[24] |
| $\mathbf{X}^R_{\text{SW}}$ | Ratio of subjective words(By sentiWordNet [64])[24] |
| $\mathbf{X}^R_{\text{Rating}}$ | Review Rating |
| $\mathbf{X}^R_{\text{Days}}$ | Review published since user's first comment |

**Table 2**
User and item features.

| Symbol | Definition |
|---|---|
| $\mathbf{X}^U_{\text{MNR}}$ | Maximum number of user's reviews written in a day [62] |
| $\mathbf{X}^U_{\text{ISR}}$ | Is user's sole review?[12] |
| $\mathbf{X}^U_{\text{PR}}$ ; $\mathbf{X}^I_{\text{PR}}$ | Ratio of user's and item's positive reviews(4–5 star)[11] |
| $\mathbf{X}^U_{\text{NR}}$ ; $\mathbf{X}^I_{\text{NR}}$ | Ratio of user's and item's negative reviews(1–2 star)[11] |
| $\mathbf{X}^U_{\text{AvgRD}}$ ; $\mathbf{X}^I_{\text{AvgRD}}$ | User's and Item's average rating deviation [63] |
| $\mathbf{X}^U_{\text{WRD}}$ | User's weighted rating deviation [63] |
| $\mathbf{X}^U_{\text{RE}}$ ; $\mathbf{X}^I_{\text{RE}}$ | User's and item's Rating Entropy [28] |
| $\mathbf{X}^U_{\text{BST}}$ | Is fraudster account established within one month?[11] |
| $\mathbf{X}^U_{\text{ER}}$ ; $\mathbf{X}^I_{\text{ER}}$ | Ratio of Extremity rating(1 and 5 star)[62] |

We apply a vector $\mathbf{s} = [x_1, x_2, \ldots, x_i]$ to represent a sentence, where $x_i$ denotes a word or punctuation, and the task of pretrained semantic parsing is to convert vector $\mathbf{s}$ to a low-dimensional vector $\mathbf{v}$, which represents the sentence's semantics.

$f : \mathbf{s} \rightarrow \mathbf{v}$, where $f$ is the pretrained language model.

**Definition 2. Syntactic Structure Parsing** is an analysis method that aims to obtain the syntactic structure or complete phrase structure of the whole sentence, also known as constituent structure parsing [61]. We adopt syntactic trees $\mathcal{T}$ and syntactic subtrees $\tau$ to recursively represent trees $t = (\mathbf{r}, [t_1, \ldots, t_k])$ where $\mathbf{r}$ is the root embedding and $t_i$ is child tree [42].

**Example.** Given a sentence like *Best hotdogs ever*, a syntactic structure parser converts sentences into the syntactic tree shown in Fig. 1, where syntactic tree $t = (S, [\text{NP}, \text{ADVP}, \text{PU}])$, syntactic subtrees $t_1 = (\text{NP}, [\text{JJS}, \text{NNS}])$, $t_{11} = (\text{JJS}, [\text{Best}])$.

**Definition 3** (*Behavior Analyze*)**.** We analyze and model the behavior of users, items, and reviews in spam review detection problems using statistical methods [11,62,63]. We convert the behaviors into feature vectors. Table 1 shows the review behavior feature vectors, and Table 2 shows the behavior feature vectors of users and items, and the features extracted in ways that reference the paper [12].

**Definition 4** (*Spam-Graph*)**.** To express that the user reviewed the item, we establish a Spam-Graph $\mathcal{G}(U, I, R, W)$ shown in Fig. 3(a),

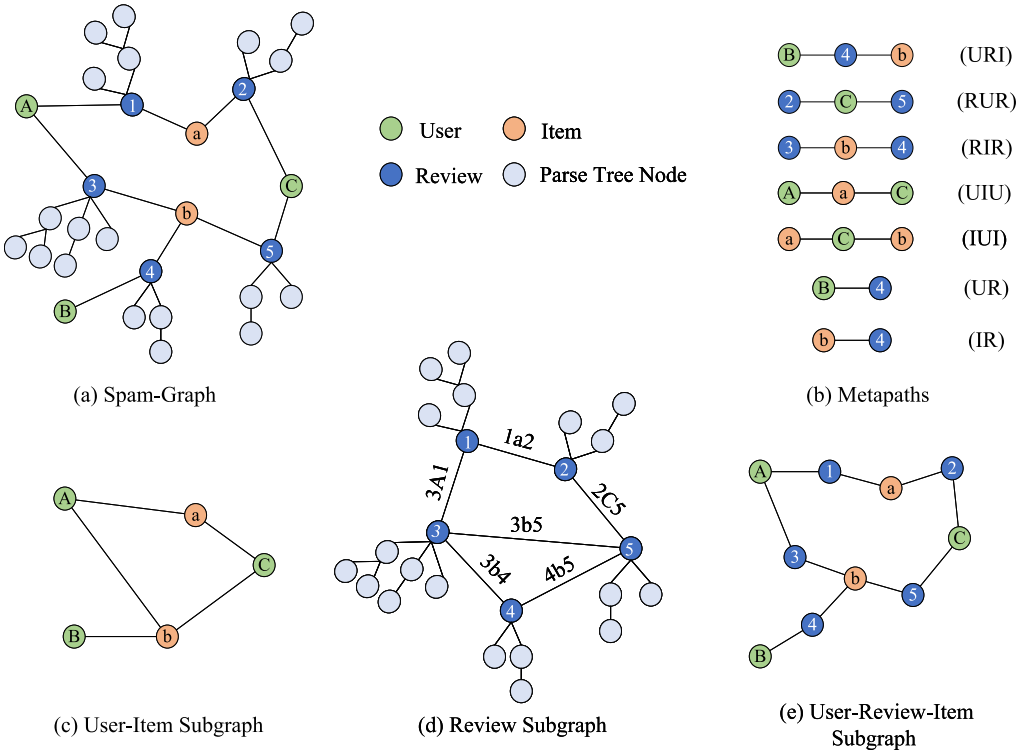


**Fig. 2.** Example of Spam-Graph, metapaths, and metapath-based subgraphs of Spam-MSANN model in Fig. 3. **(a) Spam-Graph:** example of Spam-Graph. **(b) Metapaths:** Instances of seven different types of metapaths, where URI, RUR, RIR, UR, IR are utilized to generate subgraphs, and UIU, IUI, URI are used for node aggregation. **(c) User-Item Subgraph** generated by metapath URI. **(d) Review Subgraph** generated by metapath RUR and RIR. **(e) User-Review-Item Subgraph** generated by metapath UR and IR.

where $u \in U$, $i \in I$ represent the user node and the item node, and $w \in W$ represents the syntactic tree nodes of the review syntax tree $\mathcal{T}$ with the root $r \in R$ as the review node.

**Definition 5** (*Metapath*)**.** Generally, a metapath is a sequence of abstract nodes alternating with concrete edges. The meta of a metapath refers to the fact that the node in the path is not a definite instance node but the abstract type represented by this node. For example, for the simplest path, which is actually a triple (I, love, China), its metapath, should be (person, love, country). In this paper, a metapath $P$ refers to a specific path in the form of $A_0 \xrightarrow{R_0} A_1 \xrightarrow{R_1} \dots \xrightarrow{R_{l-1}} A_l$ in graph, where $A_l$ and $R_l$ corresponds to the node type and relation on $P$. Specifically, we apply $A_0A_1 \dots A_l$ to abbreviate the metapath [49,65].

**Example.** Fig. 2(b) depicts seven metapaths: URI, RUR, RIR, UIU, IUI, UR, and IR, where syntactic tree nodes are stored in the review node and are not engaged in the metapath setting. We divided these seven metapaths into two categories: subgraph generation, shown in Fig. 3(b), and node aggregation, shown in Fig. 3(c). Take the $U \xrightarrow{R_0} R \xrightarrow{R_1} I$(URI) as an example. URI represents the action of one user ($U$) posting a review ($R$) on an item ($I$), while $R_0$ and $R_1$ denote the relationships where the user posts a review and the item receives a review, respectively.

**Definition 6** (*Metapath-based Subgraph*)**.** Given a metapath $P$ of a Spam-Graph $\mathcal{G}$, the metapath-based subgraph $\mathcal{G}^P$ is a graph constructed by node $v$ and metapath-based neighbors $\mathcal{N}_v^P$, where $\mathcal{N}_v^P$ is the set of nodes that connect to node $v$ via metapath $P$[48].

**Example.** Considering a metapath instance $P$ = URI in Fig. 2(b), user B is a metapath-based neighbor $\mathcal{N}_b^{\text{URI}}$ of item b, where review node 4 is an intermediate node along the specific metapath $P$. Then metapath-based User-Item subgraph $\mathcal{G}^{\text{URI}}$ in Fig. 2(c) is constructed by all the metapath-based neighbor $\mathcal{N}_v^{\text{URI}}$ pair in Spam-Graph $\mathcal{G}$.

**Spam Detection Problem Definition.** Given a Spam-Graph $\mathcal{G}(U, I, R, W)$ and review truthfulness label $y_r^{\text{label}}$, Spam-MSANN algorithm infers the truthfulness of reviews based on graph structure and node information.

## 4. Methodology

In this section, details of our Spam-MSANN are presented, and the overall architecture is shown in Fig. 3. The model architecture contains four parts, which are (a) Spam-Graph Construction, (b) Metapath-based Subgraph Generation, (c) Subgraph Representation and Node Aggregation, and (d) Subgraph Aggregation and Classification. In part (a), the dashed rectangle shows how to build a syntactic tree for the example sentence in Definition 2.

### 4.1. Spam-graph construction

To illustrate the fact that the user reviewed the item, we construct a Spam-Graph $\mathcal{G}(U, I, R, W)$ where $u \in U$, $i \in I$ represent user node and item node, and we use $w \in W$ to represent the syntactic tree nodes of the review syntactic tree $\mathcal{T}$ with the root node $r \in R$ as the review node. If user A posts a review B on item C, there are two relation edges between A, B and B, C, and the syntactic tree $\mathcal{T}_B$ in the graph stores the syntactic structure of review B. We employ syntactic structure parser such as Stanford CoreNLP [66] to convert review B into syntactic tree $\mathcal{T}_B$, and the details of $\mathcal{T}_B$ are shown in the bottom dotted box in Fig. 3(a).

In Spam-Graph, different metapaths can reveal different information. Setting metapaths manually and learning the semantics

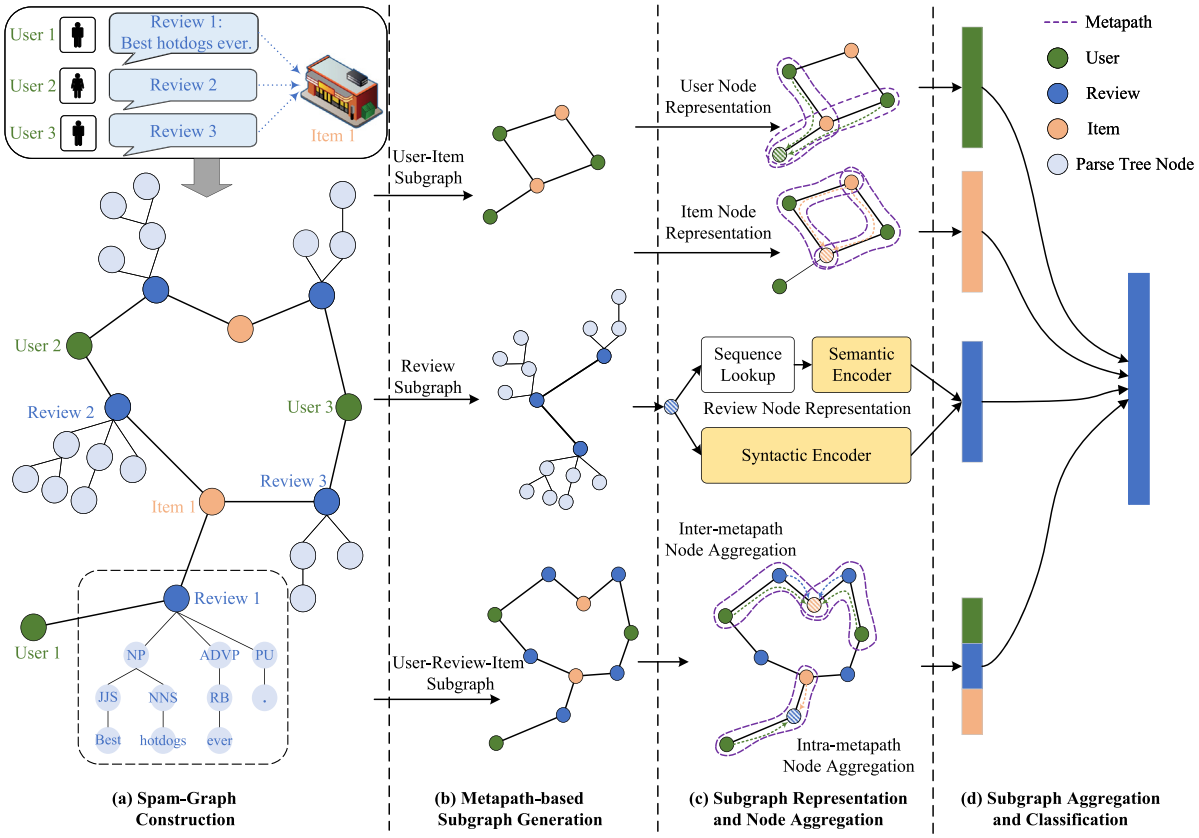


**Fig. 3.** An overview of Spam-MSANN model. **(a) Spam-Graph Construction:** Spam-Graph is constructed by four different types of nodes and interrelationship edges among nodes. The solid rectangle shows a partial example of Spam-Graph with three users reviewing the same item. The syntactic tree of Review 1 written by User 1 is shown in the dashed rectangle. **(b) Metapath-based Subgraph Generation:** The three subgraphs are generated by the metapaths defined in Fig. 2. **(c) Subgraph Representation and Node Aggregation:** This section shows the subgraph representation and node aggregation approaches based on different metapaths, where the dashed part represents a specific metapath. The shaded nodes and dashed arrows show which nodes need to be updated and how information flows in the metapath. **(d) Subgraph Aggregation and Classification:** This section shows the subgraph aggregation and classification method, where the review node has a spam/benign label and the subgraph classification is built on the review node.

of metapaths in the subgraphs becomes vital to learn graph information. For example, metapaths can well describe the behavior of users purchasing items and can ignore elaborately camouflaged nodes when different camouflage appears. It enables our model to better detect anomalies and detect spam reviews.

In this work, we manually set up seven types of metapaths shown in Fig. 2(b) and divided them into two categories for subgraph generation and node aggregation, respectively. The details are shown in the following two sections.

### 4.2. Metapath-based subgraph generation

Intuitively, we set up five metapaths on the Spam-Graph, including URI, RUR, RIR, UR, and IR, and then we formed three subgraphs based on these metapaths. Fig. 2(b) depicts these five metapaths and Fig. 2(c),(d),(e) depicts three metapath-based subgraphs. Remarkably, the different subgraphs are independent of each other.

The construction of the subgraph is shown in Definition 6, and we next describe the specific details and differences between the construction subgraphs. We learn node information and graph structure for each of these subgraphs and verify the correctness of our intuition in the experiment section. For each of the three camouflage methods mentioned in Introduction, we build three metapath-based subgraphs to detect each of them.

- For users or items that only camouflaging reviews without camouflaging behaviors, we detect this camouflage method in User-Item subgraph generated by node $u \in U$, $i \in I$ and metapaths URI, IRU. Even if different users with different behaviors may post the same review, the review may have different authenticity. For example, suppose an item seller hires multiple user accounts to post spam reviews that are well-written or plagiarized. In that case, we must remove the impacts of the reviews, which are difficult to detect even for experts, and instead concentrate on the item to detect the abnormal behavior.
- Similarly, for users or items that only camouflage behaviors without camouflaging reviews, we detect this camouflage method in the review subgraph generated by metapaths RUR and RIR. In the graph construction, we treat both metapaths as the same, and we store the syntactic tree $\mathcal{T}$ as well as syntactic tree nodes $w$ in the review subgraph for use in the review representation section. The writing style of one user has always been stable for a long time, which means that the reviews for different items from the same user should have a similar syntactic structure. We also believe that the reviews for the same item should be similar in many aspects, such as the item description, price, quality, etc. In this case, we detect semantic and syntactic features in the review subgraph and ignore the influence of camouflaged behavior.
- For users or items whose behaviors and reviews are camouflaged at the same time, we consider them together and detect this camouflage method in User-Review-Item subgraph generated by nodes $u \in U$, $i \in I$, and metapaths UR, IR.

### 4.3. Subgraph representation and node aggregation

In this section, for each of the three subgraphs, we apply different subgraph representation and node aggregation methods to detect different camouflage.

#### 4.3.1. User-Item subgraph representation

For behavioral camouflage, we come up with a node information learning method on the User-Item subgraph in this section, and previous work [67] has demonstrated that modeling metapaths on the heterogeneous graph is effective. In User-Item bipartite graph $\mathcal{G}^{\text{URI}}$, we apply two types of metapaths, $P_1$ = UIU and $P_2$ = IUI, to detect the camouflage behavior of users and items respectively, and then we refer to metapath2vec [49] for graph embedding. Metapath2vec presents a metapath-based random walk strategy to walk on the graph from a starting node $v \in V$ in $\mathcal{G}^{\text{URI}}$ to a heterogeneous context $N_t(v)$ according to the transition probability $p$, where $N_t(v)$ denotes $v$'s neighborhood with the $t$th type of nodes in metapath $P$. Metapath2vec then adapts the heterogeneous Skip-gram algorithm to obtain starting nodes' embeddings by maximizing the probability of nodes' heterogeneous context based on metapath $P = (P_1, P_2)$. The optimal strategy is calculated as follows:

$$\arg\max_{\theta} \sum_{v \in V} \sum_{t \in T_V} \sum_{c_t \in N_t(v)} \log p\left(c_t \mid v; \theta\right) \tag{1}$$

where the random walk transition probability $p$ and the negative sampling-based optimization strategy of Eq. (1) is described in [34,49], separately.

Finally, we obtain the representation of nodes $u$ and $i$ in User-Item subgraph $\mathcal{G}^{\text{URI}}$, and the calculation is as follows, where $\mathbf{x}_u^{\text{UIGraph}}$ and $\mathbf{x}_i^{\text{UIGraph}}$ are the representations of user node and item node, respectively.

$$\begin{aligned} \mathbf{x}_u^{\text{UIGraph}} &= \text{Metapath2vec}\left(u \in U; \mathcal{G}^{\text{URI}}, P_1 = \text{UIU}\right) \\ \mathbf{x}_i^{\text{UIGraph}} &= \text{Metapath2vec}\left(i \in I; \mathcal{G}^{\text{URI}}, P_2 = \text{IUI}\right) \end{aligned} \tag{2}$$

#### 4.3.2. Review subgraph representation

Previous studies have shown that the writing style of the same user is stable, so we assume that the syntactic structure in the reviews of this user is similar even if he reviews different items. Also, we think that reviews for the same item have similar semantic relevance. For example, these reviews also talk about the item's price, quality, and whether or not the user would repurchase it.

• **Sampling Method and Training Percentages.** In review subgraph $\mathcal{G}^P$ where $P$ = (RUR, RIR), we design a graph sampling method [68] under different training percentages, where the syntactic trees $\mathcal{T}$ as well as nodes $w \in W$ are stored in the review node and does not participate in sampling. To investigate the effect of different training percentages on model detectability, we randomly sampled an equal number of benign and spam review nodes separately under different training percentages. Then, for all the sampled review nodes in the training set, we re-order them and input them into the encoders below.

Specifically, we randomly select a fixed number of review nodes $r \in R$ and sample a fixed number of $S$ neighbor nodes of $r$. Note that the selected neighbor nodes are still in the training set, so that test labels are not introduced. If the number of $r$'s neighbors is less than $S$, we adapt a resampling strategy with put-back until $S$ nodes are sampled. Then we input the neighbors of node $r$ and $r$ itself into the syntactic encoder and semantic encoder to learn the syntactic and semantic context for users and items. Then we utilize the well-trained syntactic encoder and semantic encoder to obtain the representation of all the review nodes $r \in R$ in the subgraph $\mathcal{G}^P$.

For example, in Fig. 2(d) we select 2 nodes and set $S = 2$. We select node 1 and node 5 as the initial nodes and randomly sample nodes 2,3 and nodes 2,4 as neighbor nodes. Then the order of input to the syntactic encoder and semantic encoder is (1, 2, 3, 5, 2, 4).

• **Syntactic Encoder**. We employ KERMIT (Kernel-inspired Encoder with a Recursive Mechanism for Interpretable Trees) [42] to encode syntactic trees $\mathcal{T}$ into neural networks. For each review node $r \in R$ in the review subgraph $\mathcal{G}^P$ where $P$ = (RUR, RIR), we learn the syntactic embeddings between different reviews. The review's syntactic tree is encoded in the following way:

$$\mathbf{y}_r^{\mathcal{T}} = \mathcal{D}(\mathcal{T}) \tag{3}$$

where $\mathcal{D}$ is KERMIT encoder and $\mathbf{y}_r^{\mathcal{T}}$ is syntactic embeddings for syntactic tree $\mathcal{T}$.

• **Semantic Encoder**. We have encoded the syntactic tree above, and next, we encode the reviews in terms of semantic parsing. Preprocessing text with word and sentence embeddings has become essential for natural language processing techniques. Recently, advanced Transformer-based [36] pre-trained language model such as BERT (Bidirectional Encoder Representations from Transformers) [37] has achieved good performance in text pre-training. These models have been trained on a large unlabeled corpus and enable better performance in downstream tasks such as review text classification [42]. However, due to computational efficiency, BERT's maximum input sequence length is 512. The average length of reviews in the dataset exceeds 512, so a lot of information is lost using BERT.

Consequently, we apply BigBird [69] proposed by Google as a semantic encoder. BigBird improves the global attention mechanism in Transformer and reduces the complexity of Transformer to linear using sparse attention mechanisms including random attention, window attention, and global attention, which enables the maximum input sequence length to 4096. Given a review node $r$, we adapt lookup decoder $\mathcal{D}_{\text{lookup}}$ to revert the syntactic tree $\mathcal{T}_r$ to the original text sequence $\mathbf{x}_r^{\text{Seq}}$, and the input review has been preprocessed and tokenized as BigBird. The lookup decoder is a mapping function that is saved during syntactic tree construction to dig the original text sequence in the syntactic tree $\mathcal{T}_r$.

The lookup decoder and BigBird encoder are represented as follows, corresponding to the Sequence Lookup module and Semantic Encoder module shown in Fig. 3(c), respectively.

$$\begin{aligned} \mathbf{x}_r^{\text{Seq}} &= \mathcal{D}_{\text{lookup}}\left(\mathcal{T}_r\right) \\ \mathbf{y}_r^{\text{Seq}} &= \text{BigBird}\left(\mathbf{x}_r^{\text{Seq}}\right) \end{aligned} \tag{4}$$

We finally input BigBird's final [CLS] token representation to the classifier, as specified in the next section.

• **Subgraph Representation**. For each batch, we input the sampled training sequences of review nodes into the syntactic and semantic encoders, respectively. Then we input the output of the syntactic and semantic encoder as well as the review's behavior features shown in Table 1 into the multi-layer perceptron (MLP) for classification:

$$\begin{aligned} y_r^{\text{pred}} &= \text{MLP}\left(\text{concat}\left(\mathbf{y}_r^{\mathcal{T}}, \mathbf{y}_r^{\text{Seq}}, \mathbf{x}_r^{\text{Bh}}\right)\right) \\ \mathbf{x}_r^{\text{Bh}} &= \text{concat}\left(\mathbf{x}_{\text{RES}}^r, \mathbf{x}_{\text{PC}}^r, \ldots, \mathbf{x}_{\text{Days}}^r\right) \end{aligned} \tag{5}$$

where the activation function of the last fully connected layer in MLP is Softmax. $\mathbf{y}_r^{\mathcal{T}}$ is the output of the syntactic encoder, $\mathbf{y}_r^{\text{Seq}}$ is the output of the semantic encoder, and $\mathbf{x}_r^{\text{Bh}}$ is the review feature vector [12] concatenated from all the features in Table 1. We input all the nodes in the review subgraph into the well-trained model and employ the parameters of the penultimate fully connected layer in MLP as feature vectors $\mathbf{x}_r^{\text{ReGraph}}$ for the review subgraph representation.

*4.3.3. User-Review-Item subgraph aggregation*

For the case of reviews and behaviors camouflaged together, we consider behaviors and reviews together as detection targets, which were introduced in GAS [15] and improve the node aggregation method based on GCN [20] and GAT [21].

Specifically, we present a heterogeneous graph convolutional networks method based on URI metapath in User-Review-Item subgraph $\mathcal{G}^{\text{URI}}$, and we treat URI and IRU as the same. For node $u \in U$, $i \in I$ and $r \in R$ on the $P_{\text{URI}}$ and $P_{\text{IRU}}$ metapaths, we utilize the node itself and the node's metapath-based neighbor nodes to update the node layer-by-layer, where $l = 1, 2, \ldots, L$ denote the number of layers in subgraph and $\mathbf{h}^l$ denote the $l$th hidden layer state of nodes. We divide the aggregation approach into user's and item's Inter-metapath aggregation and review's Intra-metapath aggregation [48].

• **Inter-metapath Aggregation**. This section introduced the layer-by-layer Inter-metapath aggregation approach for the hidden states of the user node $\mathbf{h}_u^l$ and the item node $\mathbf{h}_i^l$. We update the user and item nodes based on the information between metapaths and the node itself. We introduce an attention mechanism in the subgraph to learn the importance of different metapaths. For nodes $u$, $i$, and metapath set $P_{\text{URI}}$, $P_{\text{IRU}}$, we establish the set of neighbor vectors based on metapath according to Definition 6, then we concatenate the neighbor node vectors in the metapaths. We establish the concatenated vectors to represent the neighbors of this metapath. The set of $P_{\text{URI}}$ and $P_{\text{IRU}}$ contains all the metapaths in the User-Review-Item subgraph.

$$\begin{aligned}\mathcal{H}_u^{l-1} &= \left\{\text{concat}\left(\mathbf{h}_r^{l-1}, \mathbf{h}_i^{l-1}\right), (r,i) \in \mathcal{N}_u^{p_m}, \forall p_m \in P_{\text{URI}}\right\} \\ \mathcal{H}_i^{l-1} &= \left\{\text{concat}\left(\mathbf{h}_r^{l-1}, \mathbf{h}_u^{l-1}\right), (r,u) \in \mathcal{N}_i^{p_n}, \forall p_n \in P_{\text{IRU}}\right\}\end{aligned} \tag{6}$$

Due to different nodes having a different number of metapath-based neighbors, the number of rows of the feature matrix of $\mathcal{H}_u$ and $\mathcal{H}_i$ is unfixed, so we need to transform the dimensionality to enable the weighted dot product attention calculation. We describe the attention calculation explicitly using the user node and $P_{\text{URI}}$ as examples, while item nodes are computed asymmetrically. We define the trainable dimensional transformation matrix ($\mathbf{W}_{Tu}$, $\mathbf{W}_{Tri}$) and dimensional transformation matrix $\mathbf{W}_m^l = [1, 1, \ldots, 1]_m$ for user node. After that, we get the dimensionally aligned hidden states $\tilde{\mathbf{h}}_u^l$ and the set of candidates' feature vectors $\tilde{\mathcal{H}}_u^l$.

$$\begin{aligned}&\tilde{\mathbf{h}}_{u-1}^{l-1} = \mathbf{W}_{Tu}^{l-1} * \mathbf{h}_u^{l-1}, \quad \tilde{\mathcal{H}}_u^{l-1} = \mathbf{W}_{Tri}^{l-1} * \mathcal{H}_u^{l-1} \\ &\tilde{\mathcal{H}}_u^l = \mathbf{A}_{ttn_u} \cdot \mathcal{H}_u^{l-1} = \sigma\left(\tilde{\mathbf{h}}_{u-1}^{l-1} \cdot \tilde{\mathcal{H}}_u^{l-1}\right) \cdot \mathcal{H}_u^{l-1} \\ &\mathbf{h}_{\mathcal{N}_u}^l = \text{ReLU}\left(\mathbf{W}_U^l \cdot \mathcal{H}_u^l\right) = \text{ReLU}\left(\mathbf{W}_U^l \cdot \mathbf{W}_m^l \cdot \tilde{\mathcal{H}}_u^l\right)\end{aligned} \tag{7}$$

Where $\sigma$ is Softmax activation function and $\mathbf{W}_U^l$ is trainable matrix. For each user node, $m$ is the number of metapaths starting from the node, and the attention score $\alpha_i$ in the attention matrix is defined as $\mathbf{A}_{ttn_u} = [\alpha_1, \alpha_2, \ldots, \alpha_m]$. The calculated function of aggregated neighbor embedding $\mathbf{h}_{\mathcal{N}_i}^l$ is same as $\mathbf{h}_{\mathcal{N}_u}^l$ in a symmetric manner. So far, the message passing method is complete, and different neighbors of each node make various contributions to spam review detection with the help of attention score.

Finally, we put forward the Inter-metapath aggregation method for nodes to propagate messages between different layers, where each node $v = u, i$ receives a message from the $v$'s previous layer and the Inter-metapath messages from its metapath-$P$-based neighbors $\mathcal{N}_v^P$. The calculation is as follows:

$$\begin{aligned}\mathbf{h}_u^l &= \text{concat}\left(\mathbf{V}_U^l \cdot \mathbf{h}_u^{l-1}, \mathbf{h}_{\mathcal{N}_u}^l\right) \\ \mathbf{h}_i^l &= \text{concat}\left(\mathbf{V}_I^l \cdot \mathbf{h}_i^{l-1}, \mathbf{h}_{\mathcal{N}_i}^l\right)\end{aligned} \tag{8}$$

where $\mathbf{V}_U^l$ and $\mathbf{V}_I^l$ denote trainable weight matrices for user node and item node, $\mathbf{h}_u^l$ and $\mathbf{h}_i^l$ are the hidden state for user and item node of $l$th layer.

• **Intra-metapath Aggregation**. In User-Review-Item subgraph, for all URI metapaths and review nodes on the metapath, we aggregate the review nodes with their information and the user and item nodes on this metapath.

For example, $\forall p_m \in P_{\text{URI}}$, user node $u_m$ and item node $i_m$ are two nodes connected to review node on the metapath $p_m$. The layer-by-layer Intra-metapath aggregation approach for review node's hidden states is described below, where $\mathbf{W}_R^l$ is trainable weight matrices of $l$th layer and $\sigma$ is ReLU activation function.

$$\begin{aligned}\mathbf{h}_r^l &= \sigma\left(\mathbf{W}_R^l \cdot \text{AGG}_R^l\left(\mathbf{h}_r^{l-1}, \mathbf{h}_u^{l-1}, \mathbf{h}_i^{l-1}\right)\right) \\ \text{AGG}_R^l\left(\mathbf{h}_r^{l-1}, \mathbf{h}_u^{l-1}, \mathbf{h}_i^{l-1}\right) &= \text{concat}\left(\mathbf{h}_r^{l-1}, \mathbf{h}_u^{l-1}, \mathbf{h}_i^{l-1}\right)\end{aligned} \tag{9}$$

Finally, we take node $v$'s $L$th hidden state $\mathbf{h}_v^L = \mathbf{x}_v^{\text{URIGraph}}$ as the output of the User-Review-Item subgraph.

*4.4. Subgraph aggregation and classification*

In this section, we build a subgraph aggregation and classification method that can be applied for any kind of camouflage to automatically detect spam reviews. The nodes' subgraph-aggregation-based representation in Spam-Graph $\mathcal{G}$ is calculated as follows:

$$\begin{aligned}\mathbf{x}_u^{\mathcal{G}} &= \text{concat}\left(\mathbf{x}_u^{\text{UIGraph}}, \mathbf{x}_u^{\text{URIGraph}}\right) \\ \mathbf{x}_r^{\mathcal{G}} &= \text{concat}\left(\mathbf{x}_r^{\text{ReGraph}}, \mathbf{x}_r^{\text{URIGraph}}\right) \\ \mathbf{x}_i^{\mathcal{G}} &= \text{concat}\left(\mathbf{x}_i^{\text{UIGraph}}, \mathbf{x}_i^{\text{URIGraph}}\right)\end{aligned} \tag{10}$$

same as Intra-metapath aggregation method in User-Review-Item subgraph, for all the URI metapath $\forall p \in P_{\text{URI}}$ in Spam-Graph $\mathcal{G}$, we transformed the spam review detection task into a review node classification task on $\mathcal{G}$. The aggregation and classification functions are as follows, where $\sigma$ is the activation function, $\mathbf{W}_R^{\mathcal{G}}$ is the trainable matrix, and $\text{AGG}_R^{\mathcal{G}}$ is the aggregation function.

$$\begin{aligned}y_r^{\text{pred}} &= \sigma\left(\mathbf{W}_R^{\mathcal{G}} \cdot \left(\text{AGG}_R^{\mathcal{G}}\left(\mathbf{x}_u^{\mathcal{G}}, \mathbf{x}_r^{\mathcal{G}}, \mathbf{x}_i^{\mathcal{G}}\right)\right)\right) \\ \text{AGG}_R^{\mathcal{G}}\left(\mathbf{x}_u^{\mathcal{G}}, \mathbf{x}_r^{\mathcal{G}}, \mathbf{x}_i^{\mathcal{G}}\right) &= \text{concat}\left(\mathbf{x}_u^{\mathcal{G}}, \mathbf{x}_r^{\mathcal{G}}, \mathbf{x}_i^{\mathcal{G}}\right)\end{aligned} \tag{11}$$

## 5. Experiments

*5.1. Datasets*

To validate the performance of Spam-MSANN model in this paper, we have conducted the validation on two datasets (i.e. YelpCHI and Amazon), respectively, and the statistical information of these datasets is shown in Table 3.

• **Yelp** is a well-known and sizeable online review site, and we tested the performance of our model on YelpCHI, which was published by [11] in 2013 and contains data on Chicago restaurants and hotels. We remove restaurants and hotels with more than 800 reviews by referring to [16].

• **Amazon review** is came up by [70]. We select reviews from the musical instrument category as the experimental data. We referenced the paper [71] by labeling reviews with more than 75% review usefulness as benign reviews and reviews with less than 25% review usefulness as spam reviews among all reviews with more than four votes. To be consistent with the paper [16], we randomly sampled 11,944 reviews from the labeled reviews as the experimental dataset. The details of the two datasets are presented in Table 3.

**Table 3**
Graph statistics of datasets.

| Dataset | User | Item | Review | |
|---|---|---|---|---|
| | #Node | #Node | #Node | Fraud% |
| YelpCHI | 29,431 | 182 | 45,954 | 14.53 |
| Amazon | 7,596 | 5,557 | 11,944 | 23.06 |

### 5.2. Experiment settings

In Spam-Graph, for the syntactic tree node, we apply the syntactic structure parser Stanford CoreNLP [66] to convert the review into a syntactic tree, then save the original sentence and the 4000-dimensional well parsed syntactic vector on the review node. The input length to CoreNLP is limited to 4000, and the excess is tail-truncated. Same as review's behavior features in Eq. (5), for the user and item nodes, we employ the behavioral features in Table 2 as the initial embedding of Spam-Graph.

$$\begin{aligned} \mathbf{x}_u^{\mathrm{Bh}} &= \mathrm{concat}\left(\mathbf{x}_{\mathrm{MNR}}^u, \mathbf{x}_{\mathrm{ISR}}^u, \ldots, \mathbf{x}_{\mathrm{ER}}^u\right) \\ \mathbf{x}_i^{\mathrm{Bh}} &= \mathrm{concat}\left(\mathbf{x}_{\mathrm{PR}}^i, \mathbf{x}_{\mathrm{NR}}^i, \ldots, \mathbf{x}_{\mathrm{ER}}^i\right) \end{aligned} \tag{12}$$

In User-Item subgraph, we set the number of walks per node to 10, the node embedding dimension to 10, and the window size to 5. Node representation is performed for user and item nodes via UIU and IUI metapaths. In the review subgraph, we set the sampling number to 2 and the output length of the syntactic encoder to 4000. We employ the Transformer-based pretrained language model BigBird-roberta-base [69] as a semantic encoder and use BERT [37] and XLNET [38] as a comparison experiment. We utilize the final 768-dimensional [CLS] token representation of the Transformer-based model and input it into 2-layer fully connected layers. The 100-dimensional parameters of the penultimate fully connected layer are used to represent the review subgraph. In User-Review-Item subgraph, we utilize $\mathbf{x}_u^{\mathrm{Bh}}$, $\mathbf{x}_i^{\mathrm{Bh}}$ to represent the initial embedding $\mathbf{h}_u^0$, $\mathbf{h}_i^0$ of users and items at the first hidden layer, and apply the review subgraph embedding to represent the initial review node $\mathbf{h}_r^0$. The dimension of the hidden layer of the graph neural network is set to 20. In training, we employ the undersampling method to randomly sample the same number of positive and negative samples. The learning rate is 0.001, and the optimization function is Adam.

In the comparison experiment, for relational heterogeneous graphs and relational homogeneous graphs, the dimension of node embedding is 64. The baseline model settings are consistent with the paper [16]. Besides, our experimental environment is Python 3.7, PyTorch 1.8.1, and DGL 0.6.1 on Windows 10 platform, using Xeon Platinum 8260 CPU, RTX 3090 GPU, and 64 GB RAM.

### 5.3. Graph construction

Our detection model is built on Spam-Graph $\mathcal{G}(U, I, R, W)$. To demonstrate the effectiveness of our approach against advanced GNN baselines, we follow the approach of previous work [16,18,54] to introduce relational heterogeneous graph, relational homogeneous graph, and user-review-item bipartite graph as comparison experiments. The graph construction methods and experimental settings of these three graphs are the same as in previous work [15,16,54]. To identify camouflaged neighbors, previous works [16,54] employ the differences in feature similarity and label similarity between the same type nodes on the relational graph. Previous works selected different types of nodes and relational edges on different datasets to make the most significant differences in feature similarity and label similarity. In the user-review-item bipartite graph, the previous work [15] exploits the review representation as edge features for detection, enabling a better description of the realistic behavior that the user reviews the item. We next illustrate these three types of graph construction methods.

- **Relational Heterogeneous Graph Construction**. On YelpCHI, we employ review manual features [12] as nodes. There are three rules for constructing edges: R-U-R: if the same person sends two reviews, an edge is constructed between two nodes, R-S-R: if the same product review has the same rating, an edge is constructed between two nodes and R-T-R: if the same product review is posted in the same month, these edges are constructed between two nodes. On Amazon, we employ user manual features [71] as nodes. There are three rules for constructing edges: U-P-U: if two users review more than one identical product, an edge is constructed between two nodes, U-S-U: if two users have more than one same star rating within one week, an edge is constructed between two nodes and U-V-U: if the similarity of review text between the two users is greater than 5%, an edge is constructed between the two nodes.
- **Relational Homogeneous Graph Construction**. To compare with advanced homogeneous methods, we simplify the multiple relationships of the relational heterogeneous graph. The construction of the relational homogeneous graph follows the following principle: all three types of edges in the relational heterogeneous graph are considered the same type.
- **User-Review-Item Bipartite Graph Construction**. We treat the user and the item as nodes and build an edge between the two nodes if the user reviews the item. We performed a two-layer graph convolution with 29,431 user nodes, 182 item nodes, and 45,954 edges on YelpCHI graph, and we performed a two-layer graph convolution with 7,569 user nodes, 5,557 item nodes, and 11,944 edges on Amazon graph.

### 5.4. Baseline models

To validate the effect of Spam-MSANN, the following models are applied as baseline models in this paper: Relational Homogeneous Graph to GCN [20], GAT [21], GraphSAGE [68] and GeniePath [72]. User-Review-Item Bipartite Graph to GraphNAS [22], GAS [15] and Policy-GNN [23]. Relational Heterogeneous Graph to RGCN [55], FdGars [19], SemiGNN [56], Graph-Consis [54] and CARE-GNN [16].

### 5.5. Evaluation metric

Since the percentage of benign and spam reviews on YelpCHI and Amazon varies widely, with spam reviews accounting for only 14.53% and 23.06% of the total reviews, we evaluate the model using ROC-AUC[73] and Recall [74] metrics like previous work [12]. Recall metrics is defined as:

$$\mathrm{Recall} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}, \tag{13}$$

where TP is True Positive, FN is False Negative. Besides, AUC refers to the area enclosed by the coordinate axis under ROC curve [75], and AUC is defined as:

$$\mathrm{AUC} = \frac{\sum_{i \in positiveClass} rank_i - \frac{M(1+M)}{2}}{M \times N}, \tag{14}$$

where $M$ is the number of positive samples and $N$ is the number of negative samples. We utilize score to denote the probability that each test sample belongs to a positive sample, and sort the scores from largest to smallest, where the *rank* of the sample corresponding to the largest score is $n$, and the *rank* of the second largest is $n-1$.

**Table 4**
Spam review detection results (%) on YelpCHI and Amazon under the different percentages of training data with different baseline models.

| Model | YelpCHI | | | | | | Amazon | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | AUC | | | Recall | | | AUC | | | Recall | | |
| | 5% | 10% | 20% | 5% | 10% | 20% | 10% | 20% | 40% | 10% | 20% | 40% |
| GCN(2016) | 54.98 | 50.94 | 53.15 | 53.12 | 51.10 | 53.87 | 75.25 | 75.13 | 74.34 | 67.81 | 66.15 | 67.45 |
| GAT(2017) | 56.23 | 55.45 | 57.69 | 54.68 | 52.34 | 53.20 | 74.55 | 72.10 | 72.16 | 65.84 | 67.13 | 65.51 |
| GraphSAGE(2017) | 53.82 | 54.20 | 56.12 | 54.25 | 52.23 | 52.69 | 73.97 | 73.97 | 75.27 | 69.36 | 70.30 | 70.16 |
| GeniePath(2018) | 56.33 | 56.29 | 57.32 | 52.33 | 54.35 | 54.84 | 72.23 | 71.89 | 72.65 | 66.63 | 65.08 | 65.41 |
| GraphNAS(2019) | 52.93 | 54.69 | 56.73 | 52.40 | 54.15 | 55.69 | 72.48 | 73.52 | 76.05 | 69.48 | 70.35 | 70.16 |
| GAS(2019) | 67.54 | 68.78 | 67.95 | 67.09 | 66.81 | 64.29 | 80.48 | 82.92 | 83.03 | 87.12 | 84.35 | 86.32 |
| Policy-GNN(2020) | 54.04 | 55.73 | 59.30 | 53.08 | 55.35 | 58.75 | 73.30 | 74.11 | 77.20 | 71.20 | 73.08 | 74.44 |
| RGCN(2018) | 50.21 | 55.12 | 55.05 | 50.38 | 51.75 | 50.92 | 74.13 | 75.58 | 74.68 | 67.22 | 65.08 | 67.68 |
| FdGars(2019) | 61.77 | 62.15 | 62.81 | 62.83 | 62.16 | 62.73 | 85.41 | 85.88 | 85.81 | 85.73 | 85.84 | 85.93 |
| SemiGNN(2019) | 53.73 | 51.68 | 51.55 | 52.28 | 52.57 | 52.16 | 76.21 | 73.98 | 70.35 | 63.32 | 61.28 | 62.89 |
| GraphConsis(2020) | 61.58 | 62.07 | 62.31 | 62.60 | 62.08 | 62.35 | 85.29 | 85.50 | 85.50 | 85.38 | 85.59 | 85.53 |
| CARE-GNN(2020) | 71.26 | 73.31 | 74.45 | 67.53 | 67.77 | 68.60 | **89.44** | **89.45** | **89.73** | 88.29 | 88.27 | **88.48** |
| **Spam-MSANN** | 70.20 | 70.56 | 71.58 | 67.74 | 68.06 | 68.81 | 80.87 | 81.71 | 84.29 | 85.27 | 83.64 | 86.22 |
| **Spam-MSANN-RS** | **72.75** | **74.03** | **74.58** | **68.14** | **69.56** | **70.22** | 86.98 | 86.14 | 87.74 | **88.62** | **89.96** | 87.01 |

## 6. Result and discussion

We perform experiments on baseline models and our Spam-MSANN model. Table 4 represents the performance of our model under the different percentages of training data, where the comparison experimental results of the baseline model are referred to the paper [16]. For all training samples in the Spam-MSANN line, we limit the sampled neighbors to the training set. In the Spam-MSANN-RS line, we randomly sample any node in the Spam-Graph to get the best review node embedding, which will introduce some test labels when we get the node's initial embedding via supervised learning. Table 6 presents the ablation experiment, and we next analyze the table results.

### 6.1. Graph construction method analysis

By comparing various graph construction methods, we demonstrate the impact of the graph's relation edges and the graph complexity on model performance. Next, we will analyze Relational Graph, User-Review-Item Bipartite Graph, and Spam-Graph.

#### 6.1.1. Relational graph analysis

We compare and analyze the relational heterogeneous graph and the relational homogeneous graph. By comparing the experimental performances of relational homogeneous graphs GCN, GAT, GraphSAGE, GeniePath, and relational heterogeneous graphs RGCN and SemiGNN, we discover that the performances of relational heterogeneous graphs are worse than those of relational homogeneous graphs. It indicates that relational heterogeneous graphs are sometimes not applicable to the problem of spam review detection, and we believe that the main reason is that these three relations are included in the user's behavioral features, such as burstiness and rating. These features work together to help make a distinction between spam and benign reviews, and splitting them into three relations is contrary to the behavioral feature extraction method in Definition 3. Nonetheless, the relational heterogeneous graph is used for spam review detection in Fdgars, GraphConsis, and CARE-GNN, as research has shown that elaborate fraudulent behavior of users can substantially impair the detector's performance. At the same time, the improvement of these works lies in sampling neighbors based on node features before aggregating them. The multiple relationships in the relational heterogeneous graph can filter out the sampling step's fraudulent behavior. For example, because there are R-T-R edges, a user who wants to lie can only post a few reviews in a month, or it will be easy to find out. These complex relationships increase the difficulty and cost of user falsification. To some extent, the sampling step eliminates the interference of some of the camouflaged neighbors, so using the relational heterogeneous graph model instead has a substantial performance improvement.

#### 6.1.2. User-Review-Item bipartite graph analysis

We compare and analyze the relational graph and User-Review-Item bipartite graph. User-Review-Item bipartite graph has thousands of nodes and tens of thousands of edges for edge classification on two datasets. In contrast, the relational graph has tens of thousands of nodes and millions of edges for node classification. The complexity of the two graphs on different datasets leads to different performances. We note that GAS outperforms FdGars and GraphConsis on YelpCHI dataset. At the same time, it is weaker than these two models on Amazon dataset, and the same phenomenon is observed with GraphNAS, Policy-GNN, RGCN, and SemiGNN. It is mainly because on Amazon, only 2.15 edges (user nodes) are connected to each item node on average, much less than the 252.49 edges (user nodes) on YelpCHI. This case results in a poorly constructed User-Review-Item bipartite graph, and the attention mechanism between nodes does not work well, either. On YelpCHI, GAS outperforms RGCN, FdGars, SemiGNN, and GraphConsis in terms of both AUC and recall metrics. This suggests that the relationship graph may be constructed in a complicated way on graphs with stronger connections between nodes, like YelpCHI, instead of Amazon. And the simpler construction of User-Review-Item bipartite graph can better detect spam reviews.

#### 6.1.3. Spam-Graph analysis

In Spam-Graph, our construction method has some relationships and differences with relational graphs and User-Review-Item bipartite graphs, which are analyzed in detail next. In User-Item subgraph, our subgraph is constructed similarly to the three edges of the relational graph R-U-R, R-T-R, and U-P-U to detect abnormal behavior between users and items. From the experimental results of Spam-MSANN and CARE-GNN, we dig that our Spam-MSANN model outperforms CARE-GNN on YelpCHI and is inferior on Amazon's AUC metric. It is because, on Amazon, our metapath-based User-Item subgraph is constructed in a way that is too sparse, with only 11,944 relational edges between the 5,557 item nodes, far fewer than the 4,398,392 edges of the relational graph [16]. Thus, the sparse graph weakens the effect of Spam-MSANN model and makes User-Item subgraph as well as the neighbor sampling method ineffective. The disadvantages of Spam-Graph's oversimplified construction method are further demonstrated compared to CARE-GNN, making our graph more

applicable to the tasks with larger data and more inter-node connections. On YelpCHI, our model achieved the best performance, proving the effectiveness of Spam-Graph.

In addition, our User-Review-Item subgraph is constructed in the same way as User-Review-Item bipartite graph. The comparison of Spam-MSANN and GAS results shows that the three subgraph aggregation is better for spam review detection than the single subgraph since Spam-Graph can detect different camouflage behaviors.

### 6.2. Training percentage analysis

To validate the performance of our model at different training percentages and to demonstrate the advantages of semi-supervised graph neural networks, we used different percentages for our experiments. To make the amount of data similar on both datasets, which is more prominent for YelpCHI than Amazon, we chose training percentages of 5%, 10% and 20% on YelpCHI, and 10%, 20% and 40% on Amazon, where the number of benign and spam reviews are each randomly sampled in half. Our chosen training percentages are consistent with previous work [16], making the experimental results comparable. In most cases, there is some improvement in the model's effectiveness as the amount of training data increases, which is consistent with our intuition.

The experimental results show that our model achieves comparable performance when the training percentage is small. This also illustrates the advantages of semi-supervised learning. We notice that in GraphSAGE, Fdgars, and GraphConsis models, the model's performance does not improve as the training percentage increases and even sometimes decreases. It indicates that the relational homogeneous graph and relational heterogeneous graphs are already complex enough to construct the graph with tens of thousands of nodes and millions of edges. So a tiny percentage of training data can already achieve the best detection performance of the model. The performance of our model improves considerably with increasing training percentage, which further proves the simplicity yet effectiveness of Spam-MSANN for constructing Spam-Graph.

### 6.3. Model complexity and computing time analysis

In this section, we analyze the model complexity and computation time of Spam-MSANN. First we evaluate the model complexity under different comparison models, where we measure the model complexity using the number of model parameters. Table 5 shows the computing times on YelpCHI of several benchmark models specifically analyzed above and the computing time of our model. The experiments of Spam-MSANN are divided into two parts: the review subgraph and the total model.

Experimental results display that CARE-GNN (1.47M parameters) outperforms Spam-MSANN in terms of model complexity (i.e. 128.00M parameters for total model and 127.95M parameters for the review subgraph) and computing time, which is mainly due to the fact that the review node embedding of the model based on the relationship graph uses statistical features rather than text embedding. BigBird pre-trained language model and the syntactic encoder occupy a large number of model parameters and computing time, but the review subgraph achieves promising performance according to ablation study. Except for the review subgraph, Spam-MSANN has some advantages in model complexity and computing time, which is consistent with the analysis of Spam-Graph in Section 6.1. This provides us the possibility to perform offline text pre-training and apply Spam-MSANN models on large-scale datasets.

**Table 5**
Models computing time on YelpCHI.

| Model | Time |
|---|---|
| GraphSAGE | 101 s |
| FdGars | 10 s |
| GraphConsis | 2,825 s |
| CARE-GNN | 132 s |
| Spam-MSANN (Review Subgraph) | 1,596 s |
| Spam-MSANN (Total) | 1,612 s |

### 6.4. Review label bias analysis

The results from both datasets in Table 4 manifest that Amazon review subgraph has better identification performance than YelpCHI subgraph, which we believe is mainly due to the inadequacy of the labeling method. On Amazon, we referenced the paper [71] by labeling reviews with more than 75% review usefulness as benign reviews and reviews with less than 25% review usefulness as spam reviews. Usefulness comes from other viewers who have viewed this product and this review, but these viewers judge the effectiveness based on the textual information. They do not know the user behavior features and item behavior features of this review and other reviews made by this user. So, labeling reviews by how useful they are is biased and gives too much weight to textual information. This is shown in the comparison results on Amazon. The same phenomenon appears in Table 6, which we introduce in the next section.

### 6.5. Ablation study

Previous work [15] took no account of the syntactic information in the context of reviews and also did not employ state-of-the-art semantic encoders for reasons such as computational efficiency. By analyzing the semantic encoder and syntactic encoder in the review subgraph, we verify our proposed hypothesis that the same user has a similar writing style (syntactic structure) and that reviews received for the same item are semantically relevant in the ablation study.

In the review subgraph, since different Transformer models have different limits on the input length, we limit the input of BERT and XLNET semantic encoders to 512 and the input of BigBird semantic encoder to 4096. The maximum input lengths of the reviews when constructing the syntax tree are also set to 512 and 4.096, respectively. Then we perform tail truncation for the exceeding part. In addition, we added a set of comparison experiments where the syntactic structure encoder is removed. At that point, the review subgraphs are identified based on semantic information and review behavior features only. The classification performances of the review subgraph are shown in Table 6, where the data in the table are the classification accuracy on the whole review graph with balanced spam/benign sampling number.

#### 6.5.1. Semantic encoder analysis

As is shown in Table 6, we discover that BigBird outperforms BERT and XLNET with or without the introduction of syntactic information. It is worth noting that the average length of reviews on YelpCHI and Amazon is 747 and 1244, respectively, both significantly longer than 512, so using BigBird semantic encoder can reduce the loss of information. By comparing the results of the four sets of experiments with XLNET and BigBird, we observe that although XLNET's semantic encoder performs better on both datasets, the performance of the semantic encoder is not sufficient to compensate for the loss of information in the review subgraph.

**Table 6**
Review subgraph detection accuracy (%) on YelpCHI and Amazon under different comparison models.

| Model | YelpCHI | Amazon |
|---|---|---|
| BERT (sqln: 512) | 65.35 | 72.13 |
| XLNET (sqln: 512) | 66.17 | 74.81 |
| BigBird (sqln: 4096) | 66.08 | 74.53 |
| BERT (sqln: 512) + Syntactic$_{ENC}$ | 67.32 | 77.99 |
| XLNET (sqln: 512) + Syntactic$_{ENC}$ | 67.26 | 80.97 |
| BigBird (sqln: 4096) + Syntactic$_{ENC}$ | **68.07** | **81.32** |

*6.5.2. Syntactic encoder analysis*

It can be seen from Table 6 that the improved performance of the review subgraph detection with the introduction of syntactic structure on both datasets demonstrates the effectiveness of our Spam-MSANN. On Amazon, the introduction of syntactic information significantly improves detection accuracy, which may likewise be the result of review bias. Viewers may prefer some neat syntax rather than judging a user's writing style, and the results of introducing syntactic structures on YelpCHI may be more in line with reality.

## 7. Conclusion and future work

This work builds a Spam-Graph and then comes up with a Spam-MSANN detection model which constructs three different subgraphs by different camouflage methods. Spam-MSANN detects these realistic and widespread spam reviews simultaneously by subgraph representation, aggregation, and classification. These subgraphs simultaneously detect spam reviews using the user's writing style, the item's review semantics, behavioral consistency, and the intrinsic relationship of users reviewing items. Experimental results display that Spam-MSANN model achieves considerable performance and contributes to the problem of spam review detection. In future work, we will integrate more knowledge (e.g., linguistic knowledge, domain knowledge, logical reasoning) into Spam-MSANN model and simultaneously optimize the proposed model's shortcomings. We will also design a more optimal sampling strategy or construct a complicated model (e.g., logical neural networks with knowledge reasoning, domain knowledge, dependent information) to optimize the sampling method and validate it on other natural language processing tasks.

## CRediT authorship contribution statement

**Zhiqiang Zhang:** Collect datasets, Funding, Supervision, Writing – review & editing. **Yuhang Dong:** Design model, Methodology, Do experiments, Writing – original draft. **Haiyan Wu:** Writing – review & editing, Supervision. **Haiyu Song:** Software, Validation. **Shengchun Deng:** Investigation. **Yanhong Chen:** Visualization.

## 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

The authors do not have permission to share data.

## Acknowledgments

This work is supported by the National Key Research and Development Program of China (No. 2019YFB1406300), the National Natural Science Foundation of China under Grant (61972336, 62073284), and Zhejiang Provincial Natural Science Foundation of China under Grant (LY22F020027, LY19F030008).

## References

[1] Yubo Chen, Jinhong Xie, Online consumer review: Word-of-mouth as a new element of marketing communication mix, Manage. Sci. 54 (3) (2008) 477–491.
[2] Kumar Ravi, Vadlamani Ravi, A survey on opinion mining and sentiment analysis: tasks, approaches and applications, Knowl.-Based Syst. 89 (2015) 14–46.
[3] Michael Luca, Georgios Zervas, Fake it till you make it: Reputation, competition, and Yelp review fraud, Manage. Sci. 62 (12) (2016) 3412–3427.
[4] Nitin Jindal, Bing Liu, Analyzing and detecting review spam, in: Seventh IEEE International Conference on Data Mining, ICDM 2007, IEEE, 2007, pp. 547–552.
[5] Sihong Xie, Guan Wang, Shuyang Lin, Philip S. Yu, Review spam detection via temporal pattern discovery, in: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2012, pp. 823–831.
[6] Michael Crawford, Taghi M Khoshgoftaar, Joseph D Prusa, Aaron N Richter, Hamzah Al Najada, Survey of review spam detection using machine learning techniques, J. Big Data 2 (1) (2015) 1–24.
[7] Atefeh Heydari, Mohammad ali Tavakoli, Naomie Salim, Zahra Heydari, Detection of review spam: A survey, Expert Syst. Appl. 42 (7) (2015) 3634–3642.
[8] Gauri Jain, Manisha Sharma, Basant Agarwal, Optimizing semantic LSTM for spam detection, Int. J. Inf. Technol. 11 (2) (2019) 239–250.
[9] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, Eduard Hovy, Hierarchical attention networks for document classification, in: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2016, pp. 1480–1489.
[10] Alper Kursat Uysal, Serkan Gunal, Semih Ergin, E. Sora Gunal, The impact of feature extraction and selection on SMS spam filtering, Elektron. Elektrotech. 19 (5) (2013) 67–72.
[11] Arjun Mukherjee, Vivek Venkataraman, Bing Liu, Natalie Glance, What yelp fake review filter might be doing? in: Seventh International AAAI Conference on Weblogs and Social Media, 2013.
[12] Shebuti Rayana, Leman Akoglu, Collective opinion spam detection: Bridging review networks and metadata, in: Proceedings of the 21th Acm Sigkdd International Conference on Knowledge Discovery and Data Mining, 2015, pp. 985–994.
[13] Jingjing Liu, Yunbo Cao, Chin-Yew Lin, Yalou Huang, Ming Zhou, Low-quality product review detection in opinion summarization, in: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL, 2007, pp. 334–342.
[14] Jiwei Li, Myle Ott, Claire Cardie, Eduard Hovy, Towards a general rule for identifying deceptive opinion spam, in: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2014, pp. 1566–1576.
[15] Ao Li, Zhou Qin, Runshi Liu, Yiqun Yang, Dong Li, Spam review detection with graph convolutional networks, in: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, 2019, pp. 2703–2711.
[16] Yingtong Dou, Zhiwei Liu, Li Sun, Yutong Deng, Hao Peng, Philip S Yu, Enhancing graph neural network-based fraud detectors against camouflaged fraudsters, in: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, 2020, pp. 315–324.
[17] Hao Peng, Ruitong Zhang, Yingtong Dou, Renyu Yang, Jingyi Zhang, Philip S Yu, Reinforced neighborhood selection guided multi-relational graph neural networks, ACM Trans. Inf. Syst. (TOIS) 40 (4) (2021) 1–46.
[18] Guan Wang, Sihong Xie, Bing Liu, S. Yu Philip, Review graph based online store review spammer detection, in: 2011 IEEE 11th International Conference on Data Mining, IEEE, 2011, pp. 1242–1247.
[19] Jianyu Wang, Rui Wen, Chunming Wu, Yu Huang, Jian Xion, Fdgars: Fraudster detection via graph convolutional networks in online app review system, in: Companion Proceedings of the 2019 World Wide Web Conference, 2019, pp. 310–316.

[20] Thomas N. Kipf, Max Welling, Semi-supervised classification with graph convolutional networks, 2016, arXiv preprint arXiv:1609.02907.
[21] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, Yoshua Bengio, Graph attention networks, 2017, arXiv preprint arXiv:1710.10903.
[22] Yang Gao, Hong Yang, Peng Zhang, Chuan Zhou, Yue Hu, Graphnas: Graph neural architecture search with reinforcement learning, 2019, arXiv preprint arXiv:1904.09981.
[23] Kwei-Herng Lai, Daochen Zha, Kaixiong Zhou, Xia Hu, Policy-GNN: Aggregation optimization for graph neural networks, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 461–471.
[24] Fangtao Huang Li, Minlie Huang, Yi Yang, Xiaoyan Zhu, Learning to identify review spam, in: Twenty-Second International Joint Conference on Artificial Intelligence, 2011.
[25] Myle Ott, Claire Cardie, Jeff Hancock, Estimating the prevalence of deception in online review communities, in: Proceedings of the 21st International Conference on World Wide Web, 2012, pp. 201–210.
[26] Mohsen Ahmadi, Moein Qaisari Hasan Abadi, A review of using object-orientation properties of C++ for designing expert system in strategic planning, Comp. Sci. Rev. 37 (2020) 100282.
[27] Geli Fei, Arjun Mukherjee, Bing Liu, Meichun Hsu, Malu Castellanos, Riddhiman Ghosh, Exploiting burstiness in reviews for review spammer detection, in: Seventh International AAAI Conference on Weblogs and Social Media, 2013.
[28] Claude E. Shannon, Prediction and entropy of printed English, Bell Syst. Tech. J. 30 (1) (1951) 50–64.
[29] Mohsen Ahmadi, Ali Taghavirashidizadeh, Danial Javaheri, Armin Masoumian, Saeid Jafarzadeh Ghoushchi, Yaghoub Pourasad, DQRE-SCnet: a novel hybrid approach for selecting users in federated learning with deep-Q-reinforcement learning based on spectral clustering, J. King Saud Univ.-Comput. Inf. Sci. (2021).
[30] Xia Hu, Jiliang Tang, Huiji Gao, Huan Liu, Social spammer detection with sentiment information, in: 2014 IEEE International Conference on Data Mining, IEEE, 2014, pp. 180–189.
[31] Yuan Gao, Maoguo Gong, Yu Xie, Alex Kai Qin, An attention-based unsupervised adversarial model for movie review spam detection, 2021, arXiv preprint arXiv:2104.00955.
[32] M. Salih Karakaşlı, Muhammed Ali Aydin, Serhan Yarkan, Ali Boyaci, Dynamic feature selection for spam detection in Twitter, in: International Telecommunications Conference, Springer, 2019, pp. 239–250.
[33] Lan You, Qingxi Peng, Zenggang Xiong, Du He, Meikang Qiu, Xuemin Zhang, Integrating aspect analysis and local outlier factor for intelligent review spam detection, Future Gener. Comput. Syst. 102 (2020) 163–172.
[34] Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean, Efficient estimation of word representations in vector space, 2013, arXiv preprint arXiv:1301.3781.
[35] Yoon Kim, Convolutional neural networks for sentence classification, Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (2014) http://dx.doi.org/10.3115/v1/D14-1181.
[36] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, Illia Polosukhin, Attention is all you need, 2017, arXiv preprint arXiv:1706.03762.
[37] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, 2018, arXiv preprint arXiv:1810.04805.
[38] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, Quoc V. Le, Xlnet: Generalized autoregressive pretraining for language understanding, 2019, arXiv preprint arXiv:1906.08237.
[39] Stefan Kennedy, Niall Walsh, Kirils Sloka, Jennifer Foster, Andrew McCarren, Fact or factitious? Contextualized opinion spam detection, 2020, arXiv preprint arXiv:2010.15296.
[40] Dan Barsever, Sameer Singh, Emre Neftci, Building a better Lie detector with BERT: The difference between truth and Lies, in: 2020 International Joint Conference on Neural Networks, IJCNN, IEEE, 2020, pp. 1–7.
[41] Noam Chomsky, Aspects of the Theory of Syntax, vol. 11, MIT Press, 2014.
[42] Fabio Massimo Zanzotto, Andrea Santilli, Leonardo Ranaldi, Dario Onorati, Pierfrancesco Tommasino, Francesca Fallucchi, KERMIT: Complementing transformer architectures with encoders of explicit syntactic interpretations, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP, 2020, pp. 256–267.
[43] Richong Zhang, Zhiyuan Hu, Hongyu Guo, Yongyi Mao, Syntax encoding with application in authorship attribution, in: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, 2018, pp. 2742–2753.
[44] Chunning Du, Jingyu Wang, Haifeng Sun, Qi Qi, Jianxin Liao, Syntax-type-aware graph convolutional networks for natural language understanding, Appl. Soft Comput. 102 (2021) 107080.
[45] Haiyan Wu, Zhiqiang Zhang, Qingfeng Wu, Exploring syntactic and semantic features for authorship attribution, Appl. Soft Comput. 111 (2021) 107815.
[46] Haiyan Wu, Zhiqiang Zhang, Shaoyun Shi, Qingfeng Wu, Haiyu Song, Phrase dependency relational graph attention network for aspect-based sentiment analysis, Knowl. Based Syst. 236 (2022) 107736.
[47] Meiling Liu, Yue Shang, Qi Yue, Jiyun Zhou, Detecting fake reviews using multidimensional representations with fine-grained aspects plan, IEEE Access (2020).
[48] Xinyu Fu, Jiani Zhang, Ziqiao Meng, Irwin King, Magnn: Metapath aggregated graph neural network for heterogeneous graph embedding, in: Proceedings of the Web Conference 2020, 2020, pp. 2331–2341.
[49] Yuxiao Dong, Nitesh V. Chawla, Ananthram Swami, Metapath2vec: Scalable representation learning for heterogeneous networks, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2017, pp. 135–144.
[50] Yizhou Sun, Jiawei Han, Mining heterogeneous information networks: principles and methodologies, Synth. Lect. Data Min. Knowl. Discov. 3 (2) (2012) 1–159.
[51] Jianan Zhao, Xiao Wang, Chuan Shi, Binbin Hu, Guojie Song, Yanfang Ye, Heterogeneous graph structure learning for graph neural networks, in: 35th AAAI Conference on Artificial Intelligence, AAAI, 2021.
[52] Xinjun Cai, Jiaxing Shang, Fei Hao, Dajiang Liu, Linjiang Zheng, HMSG: Heterogeneous graph neural network based on metapath subgraph learning, 2021, arXiv preprint arXiv:2109.02868.
[53] Tingting Liang, Xuan Sheng, Li Zhou, Youhuizi Li, Honghao Gao, Yuyu Yin, Liang Chen, Mobile app recommendation via heterogeneous graph neural network in edge computing, Appl. Soft Comput. 103 (2021) 107162.
[54] Zhiwei Liu, Yingtong Dou, Philip S. Yu, Yutong Deng, Hao Peng, Alleviating the inconsistency problem of applying graph neural network to fraud detection, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 1569–1572.
[55] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, Max Welling, Modeling relational data with graph convolutional networks, in: European Semantic Web Conference, Springer, 2018, pp. 593–607.
[56] Daixin Wang, Jianbin Lin, Peng Cui, Quanhui Jia, Zhen Wang, Yanming Fang, Quan Yu, Jun Zhou, Shuang Yang, Yuan Qi, A semi-supervised graph attentive network for financial fraud detection, in: 2019 IEEE International Conference on Data Mining, ICDM, IEEE, 2019, pp. 598–607.
[57] Yufan Zeng, Jiashan Tang, Rlc-gnn: An improved deep architecture for spatial-based graph neural network with application to fraud detection, Appl. Sci. 11 (12) (2021) 5656.
[58] Derek Lim, Xiuyu Li, Felix Hohne, Ser-Nam Lim, New benchmarks for learning on non-homophilous graphs, 2021, arXiv preprint arXiv:2104.01404.
[59] Luke S. Zettlemoyer, Michael Collins, Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars, 2012, arXiv preprint arXiv:1207.1420.
[60] Rohit J. Kate, Yuk Wah Wong, Raymond J. Mooney, et al., Learning to transform natural to formal languages, in: AAAI, vol. 5, 2005, pp. 1062–1068.
[61] Vasin Punyakanok, Dan Roth, Wen-tau Yih, The importance of syntactic parsing and inference in semantic role labeling, Comput. Linguist. 34 (2) (2008) 257–287.
[62] Arjun Mukherjee, Abhinav Kumar, Bing Liu, Junhui Wang, Meichun Hsu, Malu Castellanos, Riddhiman Ghosh, Spotting opinion spammers using behavioral footprints, in: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2013, pp. 632–640.
[63] Ee-Peng Lim, Viet-An Nguyen, Nitin Jindal, Bing Liu, Hady Wirawan Lauw, Detecting product review spammers using rating behaviors, in: Proceedings of the 19th ACM International Conference on Information and Knowledge Management, 2010, pp. 939–948.
[64] Steven Bird, Ewan Klein, Edward Loper, Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit, O'Reilly Media, Inc., 2009.
[65] Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S. Yu, Tianyi Wu, PathSim: Meta path-based top-k similarity search in heterogeneous information networks, Proc. VLDB Endow. 4 (11) (2011) 992–1003.
[66] Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, David McClosky, The stanford corenlp natural language processing toolkit, in: Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, 2014, pp. 55–60.

[67] Yizhou Sun, Brandon Norick, Jiawei Han, Xifeng Yan, Philip S. Yu, Xiao Yu, Pathselclus: Integrating meta-path selection with user-guided object clustering in heterogeneous information networks, ACM Trans. Knowl. Discov. Data (TKDD) 7 (3) (2013) 1–23.

[68] William L. Hamilton, Rex Ying, Jure Leskovec, Inductive representation learning on large graphs, 2017, arXiv preprint arXiv:1706.02216.

[69] Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al., Big bird: Transformers for longer sequences, in: NeurIPS, 2020.

[70] Julian John McAuley, Jure Leskovec, From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews, in: Proceedings of the 22nd International Conference on World Wide Web, 2013, pp. 897–908.

[71] Shijie Zhang, Hongzhi Yin, Tong Chen, Quoc Viet Nguyen Hung, Zi Huang, Lizhen Cui, GCN-based user representation learning for unifying robust recommendation and fraudster detection, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 689–698.

[72] Ziqi Liu, Chaochao Chen, Longfei Li, Jun Zhou, Xiaolong Li, Le Song, GeniePath: Graph neural networks with adaptive receptive paths, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, 2018, http://dx.doi.org/10.1609/aaai.v33i01.33014424.

[73] Tom Fawcett, An introduction to ROC analysis, Pattern Recognit. Lett. 27 (8) (2006) 861–874.

[74] Jean M. Mandler, Nancy S. Johnson, Remembrance of things parsed: Story structure and recall, Cogn. Psychol. 9 (1) (1977) 111–151.

[75] Jorge M. Lobo, Alberto Jiménez-Valverde, Raimundo Real, AUC: a misleading measure of the performance of predictive distribution models, Global Ecol. Biogeogr. 17 (2) (2008) 145–151.