Relational Graph Convolutional Networks for Social Network Analysis

Mert Bayraktar
Department of Computer Engineering Akdeniz University Antalya, Turkey 202151075008@ogr.akdeniz.edu.tr

***Abstract*—Graph-based learning has emerged as a powerful paradigm for analyzing and understanding complex relationships in various domains. Relational Graph Convolutional Networks (RGCNs) are a class of graph neural networks specifically designed to capture the rich relational information present in graph-structured data. In this report, we introduce the architecture, training procedure, and experimental results of RGCNs, showcasing their effectiveness in modeling complex relationships and achieving state-of-the-art performance in graph-related tasks.**

I. Introduction

Social networks have become an integral part of our daily lives, enabling seamless communication, information sharing, and networking opportunities. However, these platforms are not immune to the presence of malicious bots that engage in automated activities, disseminate misinformation, and engage in spamming or phishing attacks. Detecting and mitigating these bots is crucial to maintaining a safe and trustworthy online environment for users. Traditional approaches to bot detection have primarily focused on rule-based heuristics or machine-learning techniques using features derived from user behavior or content. However, these methods often suffer from limitations in capturing the complex and dynamic nature of bot strategies, leading to lower detection accuracy and increased false positives. As bots continue to evolve and adapt, it is imperative to develop advanced methodologies that can effectively identify and combat these threats. Graph-based learning provides a versatile framework for handling structured data, where entities are represented as nodes and their relationships as edges in a graph. Traditional graph convolutional networks (GCNs) have shown success in capturing local and global structural information from graphs. However, they often fail to effectively model complex relationships and heterogeneity present in real-world datasets. RGCNs overcome these limitations by explicitly incorporating relational information into the learning process. In this paper, we present the architecture of RGCNs, which extend the GCN model by introducing relation-specific parameters and operations. RGCNs enable the learning of relation-specific filters that capture different types of relationships between nodes. We discuss the aggregation and update rules used in RGCNs, which allow nodes to propagate information through the graph while considering the specific relations associated with each edge.

II. Related Works

In [1], authors focuses on the problem of Twitter bot detection and introduces a new benchmark dataset called TwiBot-20. The goal is to identify Twitter bots that engage in malicious activities and pose a threat to online discourse. The paper defines the Twitter bot detection task as a binary classification problem, where each user is classified as either human or bot.

The TwiBot-20 dataset is constructed through a data collection process that involves selecting Twitter users, retrieving multi-modal user information, and deriving trustworthy annotations. The user selection strategy employs a breadth-first search algorithm starting from different seed users, which helps in diversifying the sampled users and better representing the Twittersphere. The dataset includes users from various interest domains, such as politics, business, entertainment, and sports.

Three aspects of user information are considered in TwiBot-20: semantic, property, and neighborhood information. Semantic information refers to user-generated natural language posts, property information includes numerical and categorical user features, and neighborhood information represents the graph structure formed by followers and followings. TwiBot-20 incorporates all retrievable user information from the Twitter API, making it a comprehensive dataset for bot detection research.

To annotate the dataset, a specialized data annotation strategy is employed. Annotators, who are active Twitter users, are provided with guidelines and examples to identify bot users based on criteria such as lack of pertinence in tweets, highly automated activity, and presence of phishing or commercial links. Crowdsourcing and additional manual steps are used to ensure the trustworthiness of the annotations.

The paper also presents data analysis results to showcase the size, information completeness, user diversity, and annotation quality of TwiBot-20. It compares TwiBot-20 with existing bot detection datasets and demonstrates that TwiBot-20 is the largest dataset to date, incorporates all three aspects of user information, and covers diversified users. The annotation quality analysis shows that the annotations align with previously proposed bot characteristics.

Furthermore, the paper evaluates the performance of various bot detection methods on TwiBot-20 and compares it with other datasets. The results indicate that TwiBot-20 presents a more challenging task for bot detection, and methods that leverage multi-modal user information tend to perform better. The dataset size study highlights the benefits of the large-scale TwiBot-20 dataset, while the user information study emphasizes the importance of incorporating diverse user information

for robust bot detection.

In conclusion, this paper introduces the TwiBot-20 benchmark dataset for Twitter bot detection. The dataset provides comprehensive user information, covers diversified users, and offers a challenging task for evaluating bot detection methods. It aims to facilitate further research in identifying and combating Twitter bots that pose a threat to online discourse.

In [2], authors of the paper presents a framework for learning representations of Twitter user accounts and applies it to the task of bot detection. The goal is to classify Twitter users as either human or bot based on their semantic information, property information, and neighborhood information.

The SATAR framework consists of four major components: a tweet-semantic sub-network, a profile-property sub-network, a following-follower sub-network, and a Co-Influence aggregator. The tweet-semantic sub-network encodes the textual information of a user's tweets using hierarchical recurrent neural networks (RNNs) and attention mechanisms. The profile-property sub-network encodes the user's profile properties, such as follower count and verification status, using fully connected layers. The following-follower sub-network models the user's relationships with their followings and followers using tweet frequency and property information. Finally, the Co-Influence aggregator captures the mutual influence between the tweet semantics, user properties, and neighborhood relationships to generate a comprehensive representation of the user. The paper also introduces a self-supervised learning approach using follower count as the training signal. The SATAR framework is trained to classify users into categories based on their follower counts, utilizing the large-scale distribution of follower counts as a learning signal.

Overall, the SATAR framework aims to learn representations of Twitter user accounts that capture their semantic, property, and neighborhood information, and use these representations for bot detection. The framework is trained in a self-supervised manner using follower count as the training signal.

In [3], authors introduces the BotRGCN methodology for Twitter bot detection, which aims to address the challenges of disguise and community by leveraging multi-modal user information and user follow relationships. The paper defines the problem of Twitter bot detection as identifying bots among users using user information such as user description, tweets, numerical and categorical properties, and neighborhood information.

To encode user features, BotRGCN adopts a strategy that combines user description, tweets, numerical properties, and categorical properties into an overall user feature vector. User descriptions are encoded using pre-trained RoBERTa, and the representations are further transformed using learnable parameters. User tweets are also encoded using RoBERTa, and the representations are averaged to obtain the representation of user tweets. Numerical and categorical properties are handled using MLPs and graph neural networks.

The BotRGCN architecture involves constructing a heterogeneous graph from the Twitter network, considering both following and follower relationships between users. Relational graph convolutional networks (R-GCNs) are applied to the heterogeneous graph to learn user representations. The initial hidden vectors for nodes in the graph are derived from user features, and multiple layers of R-GCN are applied to propagate information through the graph. The user representations are further transformed using MLPs.

For learning and optimization, a softmax layer is applied to conduct Twitter bot detection based on the user representations derived from R-GCN. The loss function is constructed to minimize the cross-entropy loss between the predicted labels and the ground truth labels. Regularization is applied to control the complexity of the model.

In summary, the BotRGCN methodology addresses the challenges of disguise and community in Twitter bot detection by leveraging multi-modal user information and user follow relationships. It employs pre-trained RoBERTa for encoding user descriptions and tweets, MLPs for numerical and categorical properties, and R-GCNs for learning user representations from the heterogeneous graph. The methodology aims to improve the accuracy of Twitter bot detection by considering diverse user information and the relationships between users in the Twitter network.

In [4], authors proposes a graph-based and heterogeneity-aware Twitter bot detection framework that leverages a heterogeneous information network (HIN) and relational graph transformers. The framework aims to improve bot detection by considering relation heterogeneity and diversified interactions between users on Twitter.

The methodology begins with the construction of the HIN, where Twitter users are represented as nodes in the graph connected by different types of edges representing various relations on Twitter. The user feature vectors are transformed using a fully connected layer to serve as initial features in the graph neural networks (GNNs).

The relational graph transformers, inspired by Transformers, operate on the HIN and incorporate attention mechanisms. They calculate query, key, and value vectors for each attention head and relation, and then model influence heterogeneity by calculating attention weights between nodes. Node representations under each relation are obtained by aggregating over node neighborhoods and attention heads. A gate mechanism is applied to ensure smooth representation learning.

Semantic attention networks are used to aggregate node representations across relations while preserving relation heterogeneity. The importance of each relation is determined by taking a global view of the HIN, and the node representations are fused using these weights.

For learning and optimization, an output layer and softmax layer are applied to the final node representations for Twitter bot detection. The model is trained with supervised annotations and a regularization term to control model complexity.

In summary, the proposed methodology constructs a HIN to represent the Twittersphere, applies relational graph transformers to learn node representations under different relations, and uses semantic attention networks to aggregate representations

across relations. The framework aims to improve Twitter bot detection by considering relation heterogeneity and diversified interactions between users.

In [5], The paper presents a framework called DA-MRG for bot detection in social networks. The methodology consists of multiple components, including a multi-relational graph generator, a user representation learning module, and domain-aware classifiers. The framework is designed to leverage the heterogeneity of relations and domains to improve bot detection performance.

The methodology starts with the construction of a multi-relational graph from the initial user features and the original graph. This graph represents the interactions between users, with different types of edges indicating different relations. The generator separates the edges based on their relations and learns features for each relational graph independently.

Next, the user representation learning module is introduced to obtain high-level representations for each user. It consists of graph embedding layers and semantic attention layers. The graph embedding layers use graph neural networks (GNNs) to obtain representations for nodes in each relational graph. The semantic attention layers fuse the representations across relations based on their importance. This module aims to capture the diverse influence of different relations on user representations.

The domain-aware classifiers are proposed to discriminate between bots and humans. Multiple classifiers are trained for different domains, considering the differences in social bots across domains. The classifiers utilize the user representations obtained from the previous module and apply softmax activation to predict the bot probability for each user.

Additionally, the paper introduces a federated learning framework to address data privacy issues. Multiple participants from different social networks contribute to the model training process. Each participant trains the model locally with its own data and uploads the trained model to a central server for aggregation. The server combines the models from all participants to obtain a global model.

Overall, the DA-MRG framework leverages multi-relational graphs, user representation learning, and domain-aware classifiers for bot detection. The federated learning framework enables collaborative training across multiple social networks. The methodology aims to improve the performance of bot detection by considering relation heterogeneity and domain differences.

III. Methodology

A. Problem Formulation

Social media platforms, such as Twitter, have become integral parts of our daily lives, connecting millions of users worldwide. However, along with the growth of these platforms, there has been an increase in the presence of Twitter bots, automated accounts that mimic human behavior. Bots can be deployed for various purposes, including spreading misinformation, engaging in spamming activities, or manipulating public opinion. Detecting and distinguishing bots from genuine users is crucial for maintaining the integrity and trustworthiness of social networks.

The identification of Twitter bots poses significant challenges due to their evolving nature and sophisticated tactics. Traditional approaches based on rule-based heuristics or simple machine learning techniques have limitations in accurately capturing the complexity of bot behavior, leading to lower detection accuracy and higher false positive rates. Moreover, bot operators continuously adapt their strategies, creating a constant battle between detection algorithms and bot developers.

To address these challenges, researchers have proposed novel methodologies and algorithms for Twitter bot detection. In this paper, we aim to provide a comprehensive overview of the state-of-the-art techniques in Twitter bot detection and highlight the advancements made in this field.

We consider the task of identifying and differentiating bots from genuine users based on available user information, such as user descriptions, tweets, numerical and categorical properties, and neighborhood relationships (followings and followers). The problem is defined within the context of social network analysis, where the focus is on leveraging user features and graph structures to develop robust and accurate bot detection algorithms.

Furthermore, we discuss the key challenges associated with Twitter bot detection, including disguise, community detection, scalability, and privacy. Disguise refers to the ability of bots to mimic human behavior and evade detection algorithms. Community detection focuses on identifying bot communities and understanding their patterns of interaction. Scalability is crucial as social networks contain millions of users, and efficient algorithms are required to handle the vast amounts of data. Lastly, privacy concerns arise due to the sensitivity of social network data, necessitating the development of privacy-preserving techniques for collaborative model training.

To address these challenges, researchers have proposed various methodologies and techniques, such as graph-based approaches, deep learning models, and ensemble methods. These approaches leverage user features, graph structures, and machine-learning algorithms to differentiate bots from genuine users. Additionally, federated learning frameworks have been introduced to facilitate collaborative model training across multiple social networks while preserving data privacy.

B. Relational Graph Convolutional Networks

Given a relational graph $G = (V, E, R)$, where V represents the set of nodes, E represents the set of edges, and R represents the set of relation types, the goal of Relational Graph Convolutional Network (RGCN) is to learn node representations that capture the relational information and encode it into a low-dimensional space. Let $X \in \mathbb{R}^{|V| \times d}$ be the input node feature matrix, where $|V|$ represents the number of nodes and d represents the dimensionality of the input features. Each row of X corresponds to the feature vector of a node.

For each relation $r \in R$, RGCN applies a relational graph convolution operation to update the node representations. The

relational graph convolution operation can be mathematically defined as follows:

$$H_r^{(l+1)} = \sigma\left(\sum_{r' \in R} A_r^l X W_r^l\right)$$

where $H_r^{(l+1)} \in \mathbb{R}^{|V| \times d}$ is the updated node representation matrix for relation r at layer $l+1$, A_r^l is the adjacency matrix for relation r at layer l, and W_r^l is the weight matrix for relation r at layer l. The $\sigma(\cdot)$ represents the activation function applied element-wise.

The adjacency matrix A_r^l is defined as:

$$A_r^l = D_r^{-1/2}(I + \tilde{A}_r)D_r^{-1/2}$$

where D_r is the diagonal degree matrix for relation r, I is the identity matrix, and $\tilde{A}_r$ is the normalized adjacency matrix for relation r.

After applying the relational graph convolution operation for each relation r, the final node representations $H^{(L+1)}$ are obtained, where L represents the total number of layers. These representations can be used for various downstream tasks, such as node classification, link prediction, or graph clustering.

Finally, a softmax function is typically applied to the node representations to obtain the predicted labels or scores for classification tasks.

The RGCN model is trained by minimizing a suitable loss function, such as cross-entropy loss, using backpropagation and gradient-based optimization algorithms.

In summary, the RGCN model leverages the relational structure of the graph to capture complex dependencies and learn informative node representations. By performing relational graph convolutions, it enables effective information propagation and aggregation across the graph, leading to improved performance in tasks involving relational data.

IV. Experimental Results

We conducted extensive experiments to evaluate the performance of the Relational Graph Convolutional Networks (RGCN) model on the Twibot20 dataset. The experiments were carried out on a system with an NVIDIA GPU for accelerated computations. The RGCN model was compared against several other graph neural network models, including GCN, GAT, GraphSage, SGC.

A. Dataset

The Twibot20 dataset, a popular dataset in social network analysis, was used for our experiments. It consists of a large collection of tweets from various Twitter bots. The dataset includes node features, edge connections, and node labels indicating whether a node is a bot or a human. We split the dataset into training, validation, and test sets with a label rate of 10%. The dataset was preprocessed and transformed into a suitable format for graph-based learning.

B. Evaluation Metrics

We employed a set of commonly used evaluation metrics to assess the performance of the RGCN model. These metrics include accuracy, precision, recall, and F1-score. Accuracy measures the overall correctness of the model predictions. Precision quantifies the proportion of correctly predicted positive instances among all instances predicted as positive. Recall calculates the proportion of correctly predicted positive instances out of all actual positive instances. F1-score is the harmonic mean of precision and recall, providing a balanced measure of the model's performance.

C. Experimental Setup

The RGCN model was implemented using the PyTorch deep learning framework. We used the AdamW optimizer with a learning rate of 0.001 and weight decay of 0.001. The models were trained for a maximum of 200 epochs, and early stopping was employed based on the validation loss to prevent overfitting. The models were trained on the training set and evaluated on the test set.

D. Results

The experimental results for the RGCN model and the comparative models are summarized in Table 1. The table presents the average performance metrics across multiple runs, including accuracy, precision, recall, and F1-score. Each model was trained multiple times with different random seeds to account for the stochastic nature of the training process. The reported values represent the mean and standard deviation across the runs.

TABLE I
Experimental Results

Round	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
GCN	67.66	67.27	67.25	67.23
GAT	72.53	71.82	72.19	71.94
GraphSAGE	81.07	80.62	80.89	80.73
SGC	68.19	67.71	67.76	67.72
RGCN	81.13	80.68	81.02	80.74

V. Conclusion

In conclusion, the experimental results demonstrate the effectiveness of the RGCN model in capturing relational dependencies within the graph structure and achieving superior performance compared to other graph convolutional models. The results highlight the importance of considering relational information in graph-based tasks and support the use of RGCN as a powerful tool for analyzing complex networks.

References

[1] S. Feng, H. Wan, N. Wang, J. Li, and M. Luo, "Twibot-20: A comprehensive Twitter bot detection benchmark," in Proceedings of the 30th ACM International Conference on Information Knowledge Management, pp. 4485–4494, 2021.

[2] S. Feng, H. Wan, N. Wang, J. Li, and M. Luo, "Satar: A self-supervised approach to twitter account representation learning and its application in bot detection," in Proceedings of the 30th ACM International Conference on Information Knowledge Management, pp. 3808–3817, 2021.

[3] S. Feng, H. Wan, N. Wang, and M. Luo, "Botrgcn: Twitter bot detection with relational graph convolutional networks," in Proceedings of the 2021 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, pp. 236–239, 2021.

[4] S. Feng, Z. Tan, R. Li, and M. Luo. "Heterogeneity aware twitter bot detection with relational graph transformers," In Proceedings of the AAAI Conference on Artificial Intelligence, volume 36, pages 3977–3985, 2022.

[5] H. Peng, Y. Zhang, H. Sun, X. Bai, Y. Li, and S. Wang, "Domain-aware federated social bot detection with multi-relational graph neural networks," in 2022 International Joint Conference on Neural Networks (IJCNN). IEEE, 2022, pp. 1–8.