National College of Ireland

Detection of Twitter bot accounts using Deep Learning Techniques

MSc Research Project
MSc in Data Analytics

Rupesh Sai Baba Chintakayala
Student ID: x21104638

School of Computing
National College of Ireland

Supervisor: Aaloka Anant

National College of Ireland
Project Submission Sheet
School of Computing

National College of Ireland

Student Name:	Rupesh Sai Baba Chintakayala
Student ID:	x21104638
Programme:	MSc in Data Analytics
Year:	2022
Module:	MSc Research Project
Supervisor:	Aaloka Anant
Submission Due Date:	01/02/2023
Project Title:	Detection of Twitter bot accounts using Deep Learning Techniques
Word Count:	6125
Page Count:	17

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	30th January 2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Detection of Twitter bot accounts using Deep Learning Techniques

Rupesh Sai Baba Chintakayala
x21104638

Abstract

Automation has become a core part of everyday work, especially on social media as posting daily updates related to news, sports, and current affairs. Using automated programs known as bots to post content on social media on behalf of humans is commonly used across all major Twitter accounts. This advancement gave rise to a new set of bots that can mimic human behavior, and act as a community making it difficult to differentiate genuine human accounts from bot accounts. The aim of this project is to use the state-of-the-art graph-based Neural Network models on the most comprehensive and largest Twitter bot detection dataset (Twibot-20) to date and use the pre-trained model RoBERTa for encoding the Twitter data and develop a deep learning model to detect bots Twitter. Extensive experiments were conducted using all the models developed and the RGCN (Relational Graph Convolutional Neural Networks) model performed better than all the other models with an accuracy of 85.8%, an f1-score of 87.21%, and an MCC of 71.39%.

1 Introduction

Social media platforms such as Facebook and Twitter have gained more popularity around the world with millions of daily active users over the past decade. With the rise in technology and freemium models of these Online Social media Networks, automated programs known as bots are developed to handle the user accounts and post content. Millions of daily active users posting about their daily life, opinion on different topics, following people they admire, and commenting on posts generate data on a large scale and the validation of the data generated at this scale is quite impossible and this provides an edge for cyborgs and scammers to attack naïve people with false information. These loopholes and advancements in technology have shaped social media as a double-edged sword that is publicly available for everyone and can be used in any way one intends.

The vast support of APIs in Twitter to develop bots and post content gave rise to Twitter bots. The genuine purpose of these bots is to post content such as news, sports updates, regional updates, etc. But some of the users such as hackers, spammers, and some illicit third-party companies take advantage of these available APIs and limits to exploit the information, spread false information, malicious URLs, post tweets with the same content to influence user opinion, spam links to trap people and pull personal information, etc. Some political parties use this platform as a tool to influence public opinion by running ad campaigns and using these bots to promote curated content to win the elections. Some industries use the platform to promote goods to boost sales by posting URLs related to the products and fake comments and reviews to scam people.

Many academic studies and research have been conducted based on bot detection in social media platforms and these studies classified the bots into different types such as social spambots, traditional spambots used to increase followers, promote fake products, URL spamming, fake retweets, spread fake news, etc (Cresci et al.; 2017; Yang et al.; 2022). These studies have proposed many different techniques to detect bots in social media such as reduced feature set, One-class classifier, and feature extraction of a single tweet, etc. (Rodríguez-Ruiz et al.; 2020; Tavares et al.; 2017; Daouadi et al.; 2020; Wei and Nguyen; 2019). The main aim of this project is to develop a Twitter bot detection model using the state-of-the-art deep learning model and compare them with the existing solutions and answer the **Research Question:** *To what extent can human and bot accounts on Twitter be identified using state-of-the-art Deep Learning Neural Networks on the largest and most comprehensive Twitter bot detection dataset?*

The remainder of this paper is structured as follows: Section 2 discusses the related work in bot detection on social networks. The Methodology is outlined in Section 3 and the Design specification is briefed in Section 4. Section 5 discusses the implementation of the proposed solution, and Section 6 deals with the comparison of different experiments and evaluation of the techniques implemented. Section 7 concludes the paper along with future work.

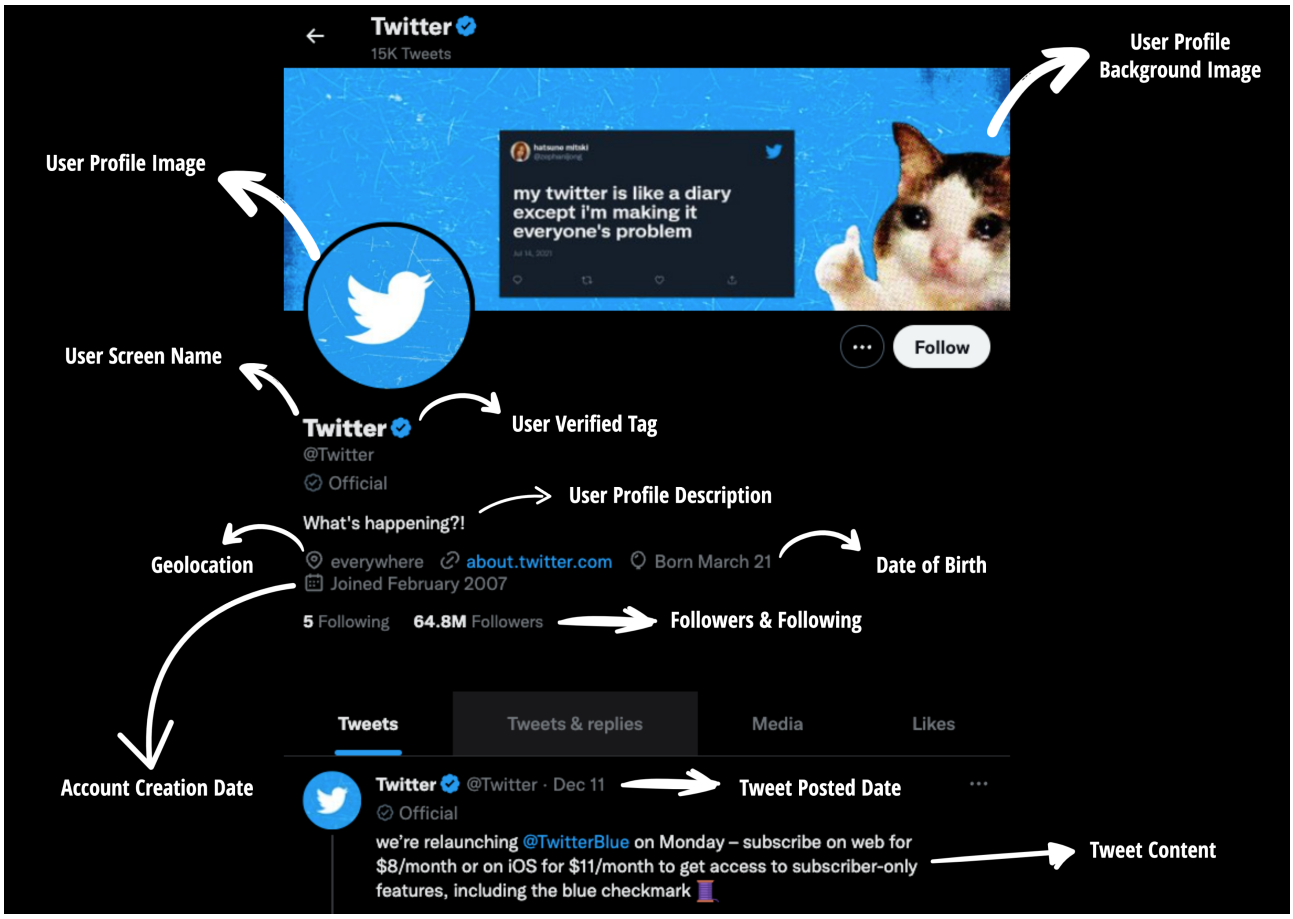

Figure 1: Twitter page

1.1 Research Question

To what extent can human and bot accounts on Twitter be identified using state-of-the-art Deep Learning Neural Networks on the largest and most comprehensive Twitter bot detection dataset?

1.2 Research Objectives

The following objectives have been observed to answer the research question:

1. Detailed analysis of all the existing methods and techniques used for bot detection.
2. Use the largest and most comprehensive Twitter bot account dataset to date to use state-of-the-art Neural Networks and develop a model to detect bots.
3. How pre-trained model RoBERTa will help in encoding the user and tweet data and help in model performance.
4. Compare general deep learning neural networks with graph-based convolutional neural networks to analyze what model will be an appropriate choice to handle this use case of detecting bot accounts on Twitter.

2 Related Work

In this section, we will discuss related work on past research and existing methods that are used for Twitter bot detection. The problem of Twitter bot detection has been researched in many different ways in the past and various techniques have been proposed. Based on past research work, the Twitter bot detection approach can be divided into different categories.

1. Content/Text-based approach
2. Feature-based approach
3. Statistical-based approach
4. Network/Graph-based Approach
5. Machine Learning & Deep Learning-based Approach

To understand the related work better, we have to understand the Twitter terminologies such as tweet, account data, tweet annotation, network structure, protected, verified, and their technical definitions.

Tweet: A short paragraph that contains up to 280 characters. Apart from text, it also supports images, mentions (tagging other Twitter users), URLs, and hashtags. The tweet data can provide a lot of insights related to the content in the tweet whether it is a spam message, malicious URL, political content spam, etc.

Account data: The account data constitutes the user information who used the Twitter account to tweet which includes information such as username, location, profile

URL, followers count, following count, account creation date, profile background, profile image URL, etc.

Tweet Annotation: Twitter processes the tweet and analyses the contextual information and provides the domain to which the tweet content matches, Twitter named it Named Entity Recognition (NER).

Network Structure: The Social network is mainly based on the network between various accounts. It helps us understand the data flow and relation between accounts. A single account or a tweet will never help in understanding the pattern, using the graph structure of multiple accounts and the tweet content the behavior and pattern can be observed.

Protected: When a user locks his profile, it cannot be accessed by anyone from the public. Duplicating/ impersonating a genuine human profile can be avoided by protecting one's profile.

Verified: When a user's identity is verified by Twitter, a blue badge is added to their name in their profile which indicates the profile is a genuine profile.

2.1 Content-based approach

The initial solution for bot detection by (Cresci et al.; 2017) was based on text analysis of the user tweets and by comparing the responses to some basic questions sent to respective users online and labeled them based on their responses. Most of the labeling was based on crowd-sourcing of the data, releasing guidelines to follow for the detection of bots, and asking the public to tag the data as a bot or human account based on the guidelines and reward them after the given tasks are performed. Human classification can be tedious in real-time implementation and moreover, the solution cannot be scaled and generalized for future purposes. However, this research has categorized Twitter accounts into 4 types they are spambots (social and traditional), fake follower accounts, and genuine human accounts. Also, the behavior of spambots was further explored and classified into different bots such as political bots which are used for retweeting political content and political campaigns, industrial bots for promoting goods and products, spamming fake URLs and malicious URLs, etc.

Many research papers proposed solutions with the same idea of text analysis like using a one-class classifier (Rodríguez-Ruiz et al.; 2020), applying sentiment analysis on tweet data of users (Dickerson et al.; 2014), etc. The one-class classifier technique only has data related to genuine human users and the model is trained based only on human behavior on Twitter, and if the account behavior is different than what is expected, they are tagged as bot accounts. This type of approach can only be used for anomaly detection, and also it can help as an initial warning system when detecting bots. But due to bots mimicking humans, this method will fail. The case is the same with (Dickerson et al.; 2014), where tweet syntax and semantics are used to analyze the tweets and proposed a framework called "Sentibot". As the complete bot behavior is unknown and analyzing only tweets to understand the behavior of bots and detect bots is not a viable option for scaling the solution.

These all techniques can be only used when the bot's behavioral pattern is similar or the motive of a bot is a single task, but due to the advancement in technology, the bots have evolved so much that they are now able to mimic humans and behave like humans. This type of approach is not used as a stand-alone approach in developing models, it is used in one of the stages of data handling such as data encoding, sentiment analysis on data for the ease of use.

2.2 Feature-based approach

Most of the techniques in recent times were based on the feature-based approach. One of them is 'Botometer', which is a publicly available tool for everyone to use to check whether any given account is showing any bot-like activity or not. Based on several features of users and tweets, it analyses the data and compares the data against many other labeled examples of Twitter users and gives a score to indicate how high the bot activity from the given account is. In (Yang et al.; 2022), they give an overview of how the Botometer works and lays the foundation for future researchers in the same path of bot detection.

Botometer was initially called Tweet BotorNot, later with the evolution of the algorithm, it was changed to Botometer. This tool was trained on many pre-existing Twitter bot account datasets. It considers over 1000 features that can be classified into different classes such as user data, user network, tweet data, language, etc. After choosing a wide range of features using machine learning algorithms, the botometer framework is developed. Similarly, the one-class classifier approach (Rodríguez-Ruiz et al.; 2020) can also be considered as a feature-based approach as well, as only one feature is chosen to develop the bot detection framework. These methods can be used to develop different techniques to detect bots, but as the behavior of bots and humans are hard to distinguish, and community of accounts comes into the picture where one needs to analyze a network of accounts to analyze the behavior of accounts and detect bots. Using only the data related to a single account and features related to it might help only at the user level but they will fail when the bots disguise themselves as humans, and the account activity is almost similar in both cases.

Few research studies had a comparative study of old techniques with the evolution of bots and suggested that the bots evolve over time learning the behavior of humans and it is hard to distinguish between them once they learn the behavior. The existing techniques should also be updated as the bots evolve and new ways should be invented to detect bots. This approach is technically not used as a whole to design bot detection models but the past work using the feature-based approach helped in understanding the importance of different features in bot detection and helps in the feature selection step of data handling when developing new models.

2.3 Statistic-based approach

Some researchers tried to use the statistical approach to perform bot detection. One of the advantages of using this approach is that it is completely independent of the content, language, and features. In (Tavares et al.; 2017), they used user posting frequency parameters to know the types the users on Twitter without knowing the content. They

classified different users as humans, bots, and cyborgs, they used the random forest algorithm to achieve an f-score of 88%. In another study (Daouadi et al.; 2020), they chose a list of user statistical features based on past work and existing approaches and used them in training a machine learning model and were able to achieve an accuracy of 95%. But these numbers are completely based on statistical analysis and this type of approach can only be used as an initial step for analysis but not completely base the solution on this. Moreover, due to the limited dataset size, this kind of approach might seem to help in bot detection but once the scale increased in real-time, the processing time increases, and the classification of bots will also fail due to many other factors involved which adds up when handling this problem of bot detection.

2.4 Network-based approach

From the previous discussions on existing methodologies and approaches to Twitter bot detection, the problem of bots disguising themselves as humans is mentioned. To handle such cases, the problem should be handled at the network level. Some authors used this approach and proposed a few techniques. One such solution is bothunter (Beskow and Carley; 2018), where the authors used Twitter user account data, tweet data, user timeline, and activity were used to create a graph to understand the interaction between accounts. One more approach (Loyola-González et al.; 2019), where the authors used a three-fold methodology that used graph analysis involving several stages for processing. Out of hundreds of features, the authors have chosen a handful of 124 features as a subset and applied NLP techniques. On top of this, different models were developed to observe the behaviour of users. A similar approach to (Loyola-González et al.; 2019), where the authors optimized the number of features and statistical and graph-based approach to build the model (Wang et al.; 2020) and achieved an accuracy of 99.94%. But all these studies were based on old botnet datasets with less scale and poor annotation quality.

2.5 Machine Learning & Deep Learning-based approach

In (Ramalingaiah et al.; 2021), the authors proposed a framework that used the bag-of-words model to detect bots on Twitter and used machine learning techniques such as Decision Tree, KNN, and Logistic regression and found out that the decision tree model is performing well with the highest accuracy compared to other models. Bag-of-words model is used for pre-processing textual data to a fixed-length vector or bag of words, this converted data is the simplest form of the textual data in the form of numbers. It mainly counts the occurrences of a word in the data and converts it to a vector. This approach is kind of similar to the text-based approach, where the solution is completely based on words and word count and there's no use of any user data or behavioral data to detect bots. In another study (Wei and Nguyen; 2019), which didn't consider any of the user data or didn't have any assumptions related to user data or graph structure, used the RNN model Bidirectional Long Short-term Memory (BiLSTM) with word embeddings based on tweet data to detect bots. No other feature is considered, simply the tweet data and their word embeddings, this is also similar to the text-based approach.

The techniques used for bot detection using machine learning and deep learning are common across all the studies where once the feature selection is performed in a feature-based approach, text data is sorted in a contentbased approach, machine learning models

are trained on top of that and they had good performance and accuracy in those studies. But the limitation across all the studies was the limited dataset size, no complete graph structure of users, and relying only on the textual data of tweets, or picking up a few features which are completely a user based but not community/ network based will not be a reliable option for real-time bot detection solutions.

3 Methodology

For this project, we follow Knowledge Discovery in Databases (KDD) methodology as our project objective is to evaluate different Deep Learning Neural Network models for graph-structured data, and KDD is considered to be the best methodology for research purposes and ends with the evaluation phase whereas CRISP-DM has deployment as the last stage and for our research, we don't have any deployment phase. Figure 2 shows an overview of the KDD methodology used for this project pictorially.

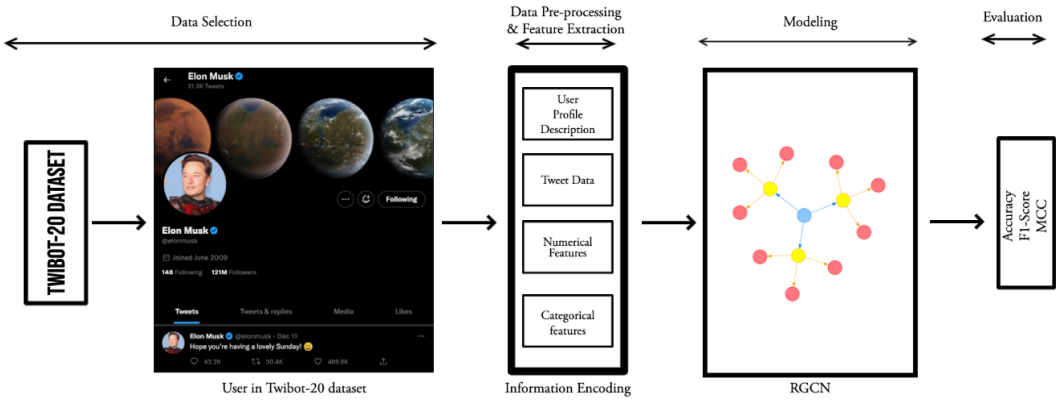

Figure 2: KDD Methodology

3.1 Data Understanding

Twibot-20 (Feng et al.; 2021) is a wide sample of Twittersphere which includes Twitter bots and genuine human users of the present generation. The sample of the dataset is available online but to get complete access, an email should be sent to the author of (Feng et al.; 2021) with the use case and details related to the project. The dataset contains four different JSON files named train, dev, test, and support with 229,573 data points. The data is partitioned in the ratio of 7:2:1 (train: validation: test) on labeled users. As the whole data is related, to preserve the graph structure, a third file has more information about users in other files. The support file provides one more unsupervised layer of data which contains information related to the neighborhood for the users in train, validation, and test data. Each user sample in the data contains:

- '**ID**' – generated by Twitter to identify the user.
- '**profile**' - which includes all the personal information obtained from Twitter API.
- '**tweet**' - which has recent tweets from a given user.

- '**neighbor**' - has 20 random followers and following of that user.
- '**domain**' – domain of the user includes business, entertainment, politics, etc.
- '**label**' – denotes whether the given user is a bot (denoted by 1) or a human (denoted by 0).

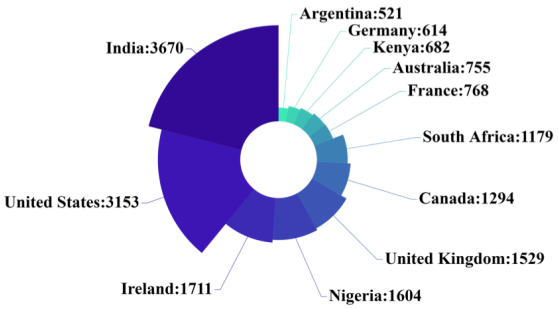

Figure 3: User location frequency in Twibot-20 country-wise

Figure 3 shows the top countries by user location from Twibot-20 dataset by frequency. The image above is taken from the research paper (Feng et al.; 2021).

3.2 Data Preparation

3.2.1 Data Pre-processing

The pre-processing of data involves feature encoding of user profile description and tweet data. Pre-trained model RoBERTa to encode the textual data, once the processing is done the output is saved as a PyTorch file. Due to the huge data size, the encoding of the data takes a lot of time and computational power. So as we run the pre-processing step once and save the output in a PyTorch file, there's no need to run these steps again. The PyTorch files which have the pre-processing output can be used as the starting point to continue with the research, this will help us train the model without taking much of our time and computational power in pre-processing steps.

3.2.2 Feature Extraction

Each user sample in the dataset contains more than 40 features including profile and tweet information. Considering all the features, the feature selection was based on numerical and categorical features which are really important in knowing about the user were considered. A feature such as a user screen name is considered and the length is calculated, similarly, the account creation date is taken and the number of days is calculated from the date of account creation. Likewise, all other attributes from the user sample from the dataset were filtered and simplified before training the model. Also, the deprecated variables by Twitter are also considered when selecting the features[1].

[1] `https://developer.twitter.com/en/docs/twitter-api/v1/data-dictionary/object-model/user`

3.3 Modeling

After analyzing past research and existing methodologies for bot detection, one of the major problems with those approaches was the dataset. The dataset used in the research was limited in size, had poor annotation quality of the dataset, and the data didn't have any graph structure. As these problems are addressed by Twibot-20 and have the graph structure with the data being highly correlated. The neural networks which support the processing of highly complex data with graph structure are Graphical Convolutional Networks (GCNs). The initial choice was to use GCNs but when the graph structure data is highly correlated, then there's an extension to GCNs which is called Relational Graph Convolutional Networks (RGCNs) (Schlichtkrull et al.; 2018) which can be used to process highly correlated graphical data. We used RGCNs to develop a model with different parameters and different attributes to observe how different attributes affect the performance of the model. Also, the best-performing model using RGCN is again developed using GCN and GAT (Graph Attention Network) to see which model is performing well with this data and bot detection use-case.

3.4 Evaluation

The evaluation metrics used to measure the performance of models that are developed using different neural networks such as RGCNs, GCNs, and GAT are accuracy, f1-score, and Matthew's correlation coefficient (MCC). MCC is a more reliable statistical measure that produces a high score only if all the quadrants in the confusion matrix have good results. MCC and f1-score are almost the same, the only difference is that if the precision and recall are very low or unknown, MCC is considered. MCC is more balanced over the assessment of classifiers and independent of the class being positive or negative.

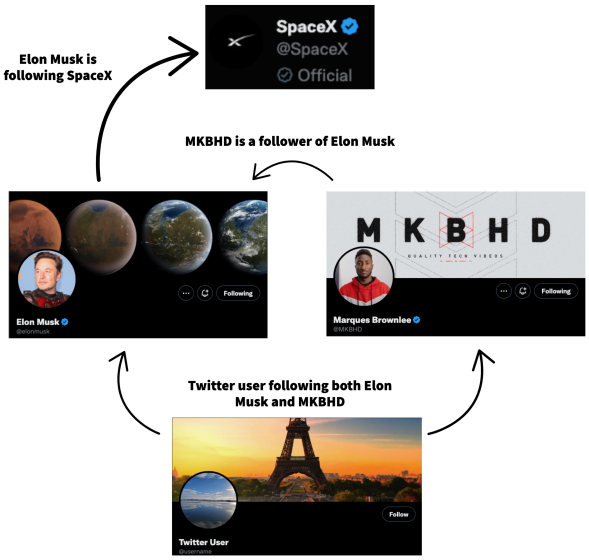

Figure 4: RGCN illustration on Twibot-20 dataset

Figure 4 shows the sample illustration of RGCN on the Twibot-20 dataset. The relation between the user nodes is either following or follower. These relations between the nodes are very much important to train the RGCN model to detect bots.

4 Design Specification

As the related work points out the limitations that exist in the old approaches used for the detection of bots on Twitter. Also, some of the Machine Learning and Deep Learning models used previously had decent accuracy and performance over the old bot detection datasets. As our dataset overcomes a few limitations and has graph-structured data, the suitable model to choose to start building a model would be Deep Learning Neural Networks which are graph-based such as GCNs, RGCNs and GATs. There are other machine-learning models which can be considered to work with the detection of bots (Ramalingaiah et al.; 2021; Wei and Nguyen; 2019), but since the graphical data is highly complex, the general machine-learning models won't be helpful in dealing with this kind of data. So, GCNs are one of the most powerful neural network architectures for graph data. But what if the graph data is relational, in our case the data is connected.

All the nodes in the graph represent Twitter users, and the edges of the graph represent the relation between the users such as followers or following. For such cases with highly relational data, RGCNs (Relational Graph Convolutional Networks) (Schlichtkrull et al.; 2018) comes into the picture. For missing information between the nodes, the authors in (Schlichtkrull et al.; 2018) used Statistical Relation Learning (SRL) and proposed two approaches link prediction and entity classification. In our case, since the relation between nodes is quite basic i.e. follower and following, we can define the relation and build a graph with the training data.
To overcome the problem of bots mimicking human behavior, the graph data is much necessary to analyze the data as a whole and stop the bot operators without giving them any chance to escape detection. To achieve this, we will be using the user information, user profile description, tweet data, and other important attributes related to the user and tweet data. We will be using RoBERTa (Liu et al.; 2019), a pre-trained model to encode the Twitter user profile description, and tweet data. Then the encoded data is converted to vectors and saved to a PyTorch file to avoid the encoding and pre-processing of data every time the code is executed. One of the advantages of using PyTorch is that once the pre-processing is done, the pre-processed files can be used to train and experiment with the models on any machine without pre-processing the data again. Moreover, the pre-processing of huge data sizes such as profile description data and mainly tweet data takes a lot of time to execute. We will be using plotly, a python library to plot graphs using the results obtained from the experiments. Also, the pipeline from transformers is used for data management while encoding the data using RoBERTa.

To summarize, Google Colab pro is used to execute the code, and the coding language used is python. PyTorch is used for ease of use and for building deep learning models. Pre-trained model RoBERTa is used for encoding the data and later PyTorch is used to convert the encoded data to vectors and store it in a PyTorch file. RGCN is used to build a model to detect bots on Twitter and this model is compared with GCN (Graph Convolutional Networks) and GAT (Graph Attention Network) models to study which model is performing well with data and use case.

5 Implementation

In this section, we will discuss the implementation of the proposed solution and models developed to detect bots using state-of-the-art deep learning models along with the libraries, parameters, and packages used to build and test the models.

- **description_size**: The size of the tensor vector of the description feature.
- **tweet_size**: The size of the tensor vector of the tweet feature.
- **numerical_properties_size**: The number of numerical features considered for building the model.
- **categorical_properties_size**: The number of categorical features considered for building the model.
- **embedding_dimension**: The size of embedding space for the vocabulary. Embedding is useful to map a vocabulary to words with the same meaning together in a low-dimensional space.
- **dropout**: Used to set the dropout rate before training the model to avoid the problem of overfitting by deep learning models.
- **lr (learning rate)**: learning rate is a hyperparameter that is used to control the response of the model w.r.t the error for every iteration where the model weights are updated. The optimal value to start the lr is 0.1 and lower it exponentially (0.01, 0.001, etc) based on the loss and training speed.

Also, there are some more parameters and functions used in developing the model. One such parameter is similar to dropout, which is weight regularization (weight_decay) which is also used to reduce the problem of overfitting in Deep Learning models. This weight regularization is applied to training data and experimenting with different values, the performance of the model is observed and the value of weight decay is finalized. We used the Leaky Rectified Linear Units (LeakyReLU) activation function which is similar to the ReLU (Rectified Linear Units) activation function but more balanced than ReLU and avoids the dead neurons problem i.e. 0 is returned as output for any input value.

6 Evaluation

In this section, we will discuss the results and performance of different deep-learning models developed using RGCN, GCN, and GAT. Each model was trained, tested, and validated using Twibot-20 data, and the model's performance was compared using accuracy (train and test), f1-score, and MCC. As discussed in the Implemantation section 5, the parameters used for all the experiments below were kept constant and experimented with the data to observe the effect of different variables on model performance. The initial setup of experiments was set up to use all the features and later, each feature is used as a stand-alone feature to train the Deep Learning Neural Network model. And then the combination of features like tweet data and user profile description and numerical and categorical features were combined.

6.1 Experiment 1 – Using all existing features

The initial setup for the experiment used all the existing data of all numerical variables and categorical variables, user profile description, and tweet data on the RGCN model, and the model was trained with 100 epochs. The main objective of this experiment is to use all the available variables after feature selection, and train the model. This sets a benchmark for the upcoming experiments to observe the performance of models using the metrics and do the comparative analysis of different models. After training, training accuracy of 89.73%, validation accuracy of 85.67%, test accuracy of 85.8%, f1-score of 87.21%, and MCC value of 71.39% were obtained.

6.2 Experiment 2 – User profile description

The next experiment was designed to use only the user profile description feature on the RGCN model. After training the model with 100 epochs, a training accuracy of 80.95%, validation accuracy of 73.83%, test accuracy of 73.2%, f1-score of 76.81%, and MCC value of 45.9% were obtained.

6.3 Experiment 3 – Tweet data

This experiment used only tweet data and the RGCN model was trained with 100 epochs. After training, a training accuracy of 79.87 %, validation accuracy of 78.1%, test accuracy of 77.01%, f1-score of 78.24%, and MCC value of 53.97% were obtained.

Comparing experiments 6.2 and 6.3, the training accuracy for the model with the description feature is more than the model with tweet data. But the final test accuracy of the model with tweet data is more than the model with description data. We can say that tweet data is more help when compared to the user description and tweet data alone for bot detection.

6.4 Experiment 4 – Numerical Features

This experiment used only numerical property features and the RGCN model was trained with 100 epochs. After training, a training accuracy of 73.33%, validation accuracy of 71.08%, test accuracy of 72.87%, f1-score of 77.75%, and MCC value of 45.98% were obtained.

6.5 Experiment 5 – Categorical Features

This experiment used only categorical property features and the RGCN model was trained with 100 epochs. After training, a training accuracy of 80.96%, validation accuracy of 79.28%, test accuracy of 81.57%, f1-score of 85.41%, and MCC value of 66.63% were obtained.

Comparing experiments 6.4 and 6.5, the training and testing accuracy for the model with the categorical feature data is more than the model with numerical feature data. From these numbers, we can infer that when the categorical and numerical features used as stand-alone features to build models for bot detection, the model with categorical data performs well with an accuracy of 81.57%.

6.6 Experiment 6 – Description + Tweets

This experiment used the user profile description and tweet data features combined and the model was trained with 100 epochs. After training, a training accuracy of 85.64%, validation accuracy of 78.86%, test accuracy of 76.75%, f1-score of 78.5%, and MCC value of 53.2% were obtained.

6.7 Experiment 7 – Numerical + Categorical Features

This experiment used the numerical properties and categorical property features combined and the model was developed using RGCN and trained with 100 epochs. After training, a training accuracy of 81.66%, validation accuracy of 79.28%, test accuracy of 81.83%, f1-score of 85.38%, and MCC value of 66.21% were obtained.

Comparing experiments 6.6 and 6.7, the training and testing accuracy for the model with the categorical and numerical features data combined is more than the model with user description and tweet data features combined. From these results, we can infer that when the model is built with categorical and numerical features combined performs better than the textual data model involving user description and tweet data, also this is the best-performing model after the initial model with all the features. This implies that the categorical and numerical features in the data are really key factors for the performance of the model.

6.8 Experiment 8 – GCN (Using all existing features)

This experiment used all the existing data of all numerical variables and categorical variables, user profile description, and tweet data, and the model was developed using GCN and trained with 100 epochs. After training, training accuracy of 78.09%, validation accuracy of 73.95%, test accuracy of 74.3%, f1-score of 76.06%, and MCC value of 48.34% were obtained.

6.9 Experiment 9 – GAT (Using all existing features)

This experiment used all the existing data of all numerical variables and categorical variables, user profile description, and tweet data, and the model was developed using GAT and trained with 100 epochs. After training, training accuracy of 80.5%, validation accuracy of 77%, test accuracy of 77.77%, f1-score of 80.53%, and MCC value of 55.2% were obtained.

Comparing experiments 6.1, 6.8, and 6.9, the RGCN model performs better than GCN and GAT models with better accuracy and MCC value. Out of these three models, GCN lacks behind in performance in terms of accuracy, f1-score, and MCC. From the results obtained from the above experiments, we can say that the best-performing model in detecting bots on Twitter is RGCN and the model with the least performance is GCN. As we have MCC which is a better measure than f1score to assess the model statistically, the RGCN model with all the features has the highest value of 71.39% which indicates that the model performing well with a good fit.

	training accuracy	validation accuracy	test accuracy	f1_score	mcc
Experiment 1	0.8973	0.8567	0.858	0.8721	0.7139
Experiment 2	0.8095	0.7383	0.732	0.7681	0.459
Experiment 3	0.7987	0.781	0.7701	0.7824	0.5397
Experiment 4	0.7333	0.7108	0.7287	0.7775	0.4598
Experiment 5	0.8088	0.7928	0.8157	0.8541	0.6663
Experiment 6	0.8564	0.7886	0.7675	0.785	0.532
Experiment 7	0.8166	0.7928	0.8183	0.8538	0.6621
Experiment 8	0.7809	0.7395	0.743	0.7606	0.4834
Experiment 9	0.805	0.77	0.7777	0.8053	0.552

Table 1: Experiment results

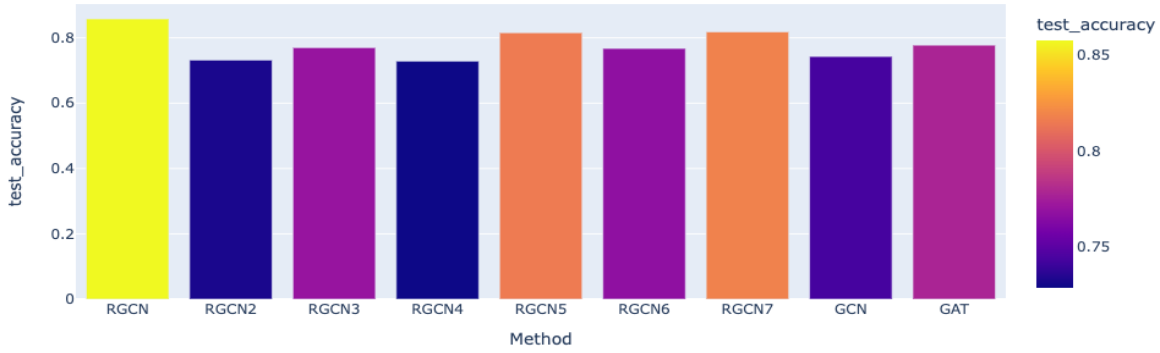

Figure 5: Test accuracy bar graph

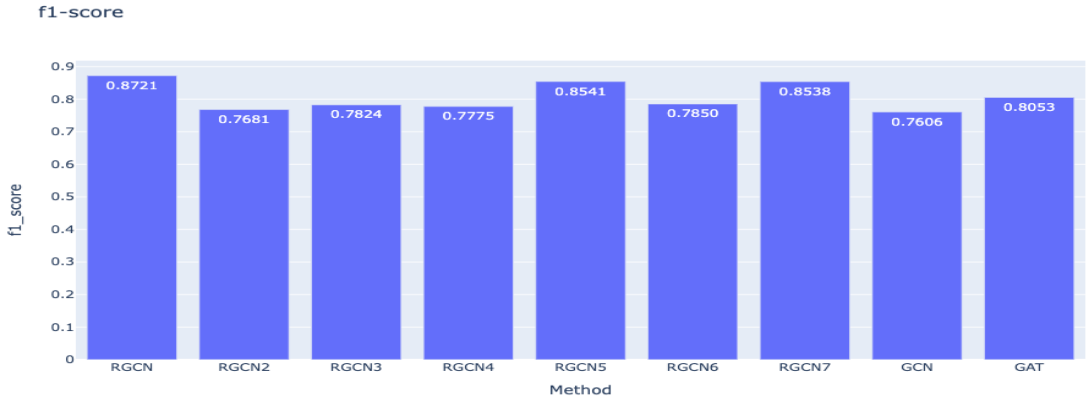

Figure 6: F1-score bar graph

6.10 Discussion

To answer the research question: *To what extent can human and bot accounts on Twitter be identified using state-of-the-art Deep Learning Neural Networks on the largest and most comprehensive Twitter bot detection dataset?*, using the results obtained from the experiments above. It was found that the RGCN model using all the existing features performs better than all other models with an accuracy of 85.8%, f1-score of 87.21%.

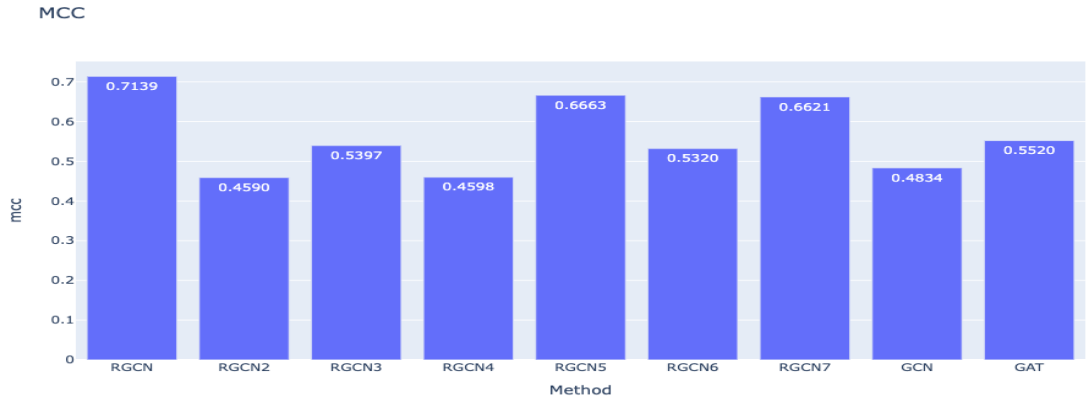

Figure 7: Matthews Correlation Coefficient bar graph

From table 1, the numbers show that the choice of RGCN over GCN for developing the model to handle the relational data was right. The model accuracies obtained for RGCN and GCN were 85.8% and 74.3% respectively. Also from the MCC values obtained, we can say that the best performing model is RGCN compared to all other models statistically with MCC value of 71.39%. Table 1 shows the results of different experiments in a tabular format with training accuracy, validation accuracy, test accuracy, f1-score, and Matthew's Correlation Coefficient. Figure 5 shows the accuracy of different models in a bar graph, and as mentioned above the RGCN model performs better than the rest of the model in the detection of bots. Figure 6 shows the f1-score bar graph for different models, we can see that there's not much variation or any trend in the score for different models, as f1-score merges precision and recall and if these are minimal or unknown then there's not much change in the value of f1-score. Whereas Figure 7 shows the bar graph with MCC values, comparing all the results from experiments, the best-performing model is RGCN with all the features and the least-performing is the RGCN model with the profile description feature after this GCN model has less MCC score indicating the GCN model is less performing compared to all other models and also RGCN and GAT.

7 Conclusion and Future Work

As the trend in the usage of social media increases and there's no validation of content posted online, there's a need to detect spam bots and bot accounts online to reduce the spread of false information. In this paper, we used the largest and most comprehensive Twitter bot dataset Twibot-20 to develop a framework for bot detection. The experiments conducted with different combinations of user information, and tweet data have shown that using all the available information on user data is necessary for the model to perform better. Using Relational Graph Convolutional Network (RGCN) over Graph Convolutional Network (GCN) and Graph Attention Network (GAT) has shown better results in terms of accuracy, f1-score, and MCC as 85.8%, 87.21%, and 71.39% respectively.

However, there are a lot of new challenges in this domain to be addressed. The evolution of bots is always there and the techniques used to detect them need to be updated with time. The end goal of this problem is to achieve a generalized solution to

detect bots and bot communities. Also with more computation power to handle huge data, behavioral annotations can be drawn for each user using their whole timeline on Twitter, and developing better models to analyze the profile along with the network data around the same user will help us get insights into every account and understand the behavior at a deeper level. Hence, there's definitely a need to explore this domain for a better future of social media networks.

8 Acknowledgment

I would like to take this opportunity to thank everyone who helped me in finishing this project on time. I'm grateful to my supervisor Mr. Aaloka Anant for his constant support, valuable input, feedback, and supervision. His constant support and feedback helped me to stay motivated and complete this project smoothly. I would like to thank all the other group members under his supervision as I believe that the constant group discussions, questioning sessions, and suggestions from all group members helped each one of us to gain knowledge over different aspects of the project and different domains. I would also like to thank the NCI library help center, for the support, and resources they provided throughout the academics, during the project implementation, and report writing. Lastly, I would like to thank my family and friends for their constant support and for motivating me throughout my academics.

References

Beskow, D. M. and Carley, K. M. (2018). Bot conversations are different: leveraging network metrics for bot detection in twitter, *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, IEEE, pp. 825–832.

Cresci, S., Di Pietro, R., Petrocchi, M., Spognardi, A. and Tesconi, M. (2017). The paradigm-shift of social spambots: Evidence, theories, and tools for the arms race, *Proceedings of the 26th international conference on world wide web companion*, pp. 963–972.

Daouadi, K. E., Rebaï, R. Z. and Amous, I. (2020). Real-time bot detection from twitter using the twitterbot+ framework., *J. Univers. Comput. Sci.* **26**(4): 496–507.

Dickerson, J. P., Kagan, V. and Subrahmanian, V. (2014). Using sentiment to detect bots on twitter: Are humans more opinionated than bots?, *2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2014)*, IEEE, pp. 620–627.

Feng, S., Wan, H., Wang, N., Li, J. and Luo, M. (2021). Twibot-20: A comprehensive twitter bot detection benchmark, *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pp. 4485–4494.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L. and Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach, *arXiv preprint arXiv:1907.11692* .

Loyola-González, O., López-Cuevas, A., Medina-Pérez, M. A., Camiña, B., Ramírez-Márquez, J. E. and Monroy, R. (2019). Fusing pattern discovery and visual analytics approaches in tweet propagation, *Information Fusion* **46**: 91–101.

Ramalingaiah, A., Hussaini, S. and Chaudhari, S. (2021). Twitter bot detection using supervised machine learning, *Journal of Physics: Conference Series*, Vol. 1950, IOP Publishing, p. 012006.

Rodríguez-Ruiz, J., Mata-Sánchez, J. I., Monroy, R., Loyola-González, O. and López-Cuevas, A. (2020). A one-class classification approach for bot detection on twitter, *Computers & Security* **91**: 101715.

Schlichtkrull, M., Kipf, T. N., Bloem, P., Berg, R. v. d., Titov, I. and Welling, M. (2018). Modeling relational data with graph convolutional networks, *European semantic web conference*, Springer, pp. 593–607.

Tavares, G., Mastelini, S. et al. (2017). User classification on online social networks by post frequency, *Anais do XIII Simpósio Brasileiro de Sistemas de Informação*, SBC, pp. 464–471.

Wang, W., Shang, Y., He, Y., Li, Y. and Liu, J. (2020). Botmark: Automated botnet detection with hybrid analysis of flow-based and graph-based traffic behaviors, *Information Sciences* **511**: 284–296.

Wei, F. and Nguyen, U. T. (2019). Twitter bot detection using bidirectional long short-term memory neural networks and word embeddings, *2019 First IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA)*, IEEE, pp. 101–109.

Yang, K.-C., Ferrara, E. and Menczer, F. (2022). Botometer 101: Social bot practicum for computational social scientists, *arXiv preprint arXiv:2201.01608* .